Opportunistic Transmission Scheduling With Resource-Sharing Constraints in Wireless Networks

Xin Liu, Edwin K. P. Chong, and Ness B. Shroff

***Abstract*—We present an "opportunistic" transmission scheduling policy that exploits time-varying channel conditions and maximizes the system performance stochastically under a certain resource allocation constraint. We establish the optimality of the scheduling scheme and also that every user experiences a performance improvement over any nonopportunistic scheduling policy when users have independent performance values. We demonstrate via simulation results that the scheme is robust to estimation errors and also works well for nonstationary scenarios, resulting in performance improvements of 20%–150% compared with a scheduling scheme that does not take into account channel conditions. Last, we discuss an extension of our opportunistic scheduling scheme to improve "short-term" performance.**

***Index Terms*—Resource allocation, scheduling, time-slotted system, time-varying channel, wireless.**

I. Introduction

In WIRELINE networks, resource allocation schemes and scheduling policies play important roles in providing quality of service (QoS), such as throughput, delay, delay-jitter, fairness, and loss rate [1]. However, resource allocation and scheduling schemes from the wireline domain cannot be directly carried over to wireless systems because of some unique characteristics in wireless channels, such as limited bandwidth, time-varying and location-dependent channel conditions, and channel-condition-dependent performance.

In wireless networks, the channel conditions of mobile users are time-varying. Radio propagation can be roughly characterized by three nearly independent phenomena: path-loss variation with distance, slow log-normal shadowing, and fast multipath-fading. Path losses vary with the movement of mobile stations. Slow log-normal shadowing and fast multipath-fading are time-varying with different time scales. Thus, users perceive time-varying service quality and/or quantity because channel conditions are time-varying. For voice users, better channel conditions may result in better voice quality. For data service, users with better channel conditions [or larger signal-to-interference-and-noise ratio (SINR)] may be served at higher data rates via adaptation techniques, such as reducing coding or spreading and/or increasing the constellation density. By using adaptation techniques, cellular spectral efficiency (in terms of b/s/Hz/sector) can be increased by a factor of two or more [2]. All the major cellular standards have included procedures to exploit this: adaptive modulation and coding schemes are implemented in the 3G TDMA standards, and variable spreading and coding are implemented in the 3G CDMA standards. In general, a user is served with better quality and/or at a higher bit rate when the channel condition is better.

On one hand, good scheduling schemes should be able to exploit the time-varying channel conditions of users to achieve higher utilization of wireless resources. On the other hand, the potential to exploit higher data throughputs in an opportunistic way, when channel conditions permit, introduces the tradeoff problem between wireless resource efficiency and levels of satisfaction among users. Because wireless spectrum is a scarce resource, improving the efficiency of spectrum utilization is important, especially to provide high-rate-data service. Hence, we cannot expect the same throughput for all users because the users in general can have very different channel conditions. However, a scheme designed only to maximize the overall throughput could be very biased, especially where there are users with widely disparate distances from the base station. For example, allowing only users close to the base station to transmit with high transmission power may result in very high throughput, but sacrifice the transmission of other users. This basic dilemma motivates our work: to improve wireless resource efficiency by exploiting time-varying channel conditions, while at the same time control the levels of fairness among users. There are various mathematical definitions of fairness in the literature [3], [4]. In this paper, we do not use a formal notion of fairness, but simply adopt the intuitive notion that no individual user should be denied access to network resources, i.e., each user is entitled to a certain amount of network resources.

We consider a time-slotted system in which time is the resource to be shared among the users. Associated with each user is a number between 0 and 1 representing the long-term fraction of time to be assigned to the user. This "time-fraction assignment" to users represents the fairness constraint that each user is granted a certain portion of the resource. The time-fraction assignment can be obtained as a byproduct of resource allocation schemes, as described in [5]. Given this resource allocation constraint, the problem is to determine which user should be scheduled to transmit at each time slot so that the network performance is optimized. To solve this problem, we present a

Manuscript received December 1, 2000; revised June 1, 2001. This work was supported in part by the National Science Foundation through Grants ECS-9501652, NCR-9624525, and ANI-9805441.

X. Liu and N. B. Shroff are with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907-1285 USA (e-mail: xinliu@ecn.purdue.edu; shroff@ecn.purdue.edu).

E. K. P. Chong was with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907-1285 USA. He is now with the Department of Electrical and Computer Engineering, Colorado State University, Fort Collins, CO 80523 USA (e-mail: echong@engr.colostate.edu).

Publisher Item Identifier S 0733-8716(01)08479-7.

scheduling scheme that maximizes the wireless resource utilization by exploiting time-varying channel conditions, taking into account the resource allocation constraint (i.e., our scheme is optimal among all schemes that satisfy the constraint). We refer to our scheme as being "opportunistic" because it takes advantage of favorable channel conditions in assigning time slots to users.

Recently, the authors of [6]–[8] have studied wireless fair scheduling policies. They extend scheduling policies for wireline networks to wireless networks that provide various degrees of performance guarantees, including short-term and long-term fairness, as well as short-term and long-term throughput bounds. However, they model a channel as either "good" or "bad," which might be too simple to characterize realistic wireless channels, especially for data service.

In [9], [10], the authors present a scheduling scheme for the Qualcomm/HDR system that satisfies the following fairness property as defined in [10]: if another scheduling algorithm is used to increase the throughput of a specific user by $x\%$ over what that user receives under the HDR scheduling algorithm, the sum of all the percentage decreases suffered by the throughputs of all the other users under the new algorithm will be more than $x\%$. This property is known as *proportional fairness* [11]. The HDR algorithm also exploits time-varying channel conditions while maintaining proportional fairness. However, their constraint is different from ours, as is their objective function. Our structure is flexible—the system can explicitly set the fraction of time assigned to each user. Furthermore, our scheme outperforms the HDR scheduler in terms of the overall throughput in all cases, although there is no guarantee that a single user performs better.

The paper is organized as follows. In Section II, we introduce the system model. In Section III, we present our opportunistic scheduling policy and its properties. We also provide a parameter estimation algorithm and discuss implementation issues. In Section IV, we use simulation results to illustrate the performance of our scheduling policy. An extension of the opportunistic scheduling algorithm to improve "short-term" performance is discussed in Section V, and conclusions are presented in Section VI.

II. System Model

We consider a time-slotted system—time is the resource to be shared among all users. A time-slotted cellular system can have more than one channel (frequency band), but at any given time, only one user can occupy a given channel within a cell. Here, we focus on the scheduling problem for a single channel. Note that a channel in this context could be very large. For example, it is possible for 10 users to share a 1-MHz frequency band for high-rate-data service, while in the IS-136 standard a voice channel takes 10-KHz bandwidth. The time-fraction assignment scheme dictates the fraction of time that a user should transmit on the channel. The scheduling algorithm then decides which time slot should be assigned to which user, given the time-fraction assignment. This time-fraction assignment can be viewed as the fairness requirement in the system that each user is entitled a certain portion of resource.

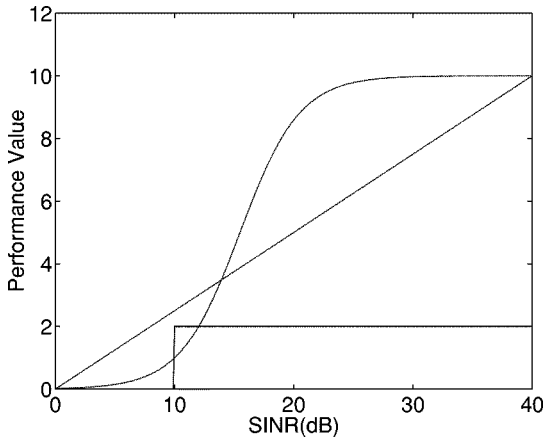

Fig. 1. Users' performance value (e.g., throughput) as a function of SINR.

As explained previously, channel conditions in wireless networks are time-varying, and thus users experience time-varying performance. We use a stochastic model to capture the *time-varying* and *channel-condition-dependent* performance of each user. Specifically, let $\{U_i^k\}$ be a stochastic process associated with user i, where U_i^k is the level of performance that would be experienced by user i if it is scheduled to transmit at time k. The value of U_i^k measures the "worth" of time slot k to the user i, and is in general a function of its channel condition. Usually, the better the channel condition of user i, the larger the value of U_i^k.

Next, we present some examples of possible performance measures. The most straightforward performance measure is the throughput (in terms of bits/sec) or the "monetary value" of the throughput (in terms of dollars/sec). Usually, a user's throughput is a nondecreasing function of its SINR. Depending on the "class" of a user, the throughput could be a step function, an S-shape function, or a linear function of the SINR, as shown in Fig. 1.

Besides throughput, other issues could also be important to users, and different users could have different performance measures. For example, power consumption is typically very important to a handset user, and hence the performance of such a user could have the form

$$\text{value of throughput} - \text{cost of power consumption}.$$

In summary, the performance value U_i^k is an abstraction used to capture the time-varying and channel-condition-dependent "worth" of a time slot to a user. In our system model, we do not make any assumptions on the physical-layer implementation of the system. The use of such a general performance model frees us from physical-layer implementation details and allows us to focus on the problem of designing scheduling policies. Different performance measures may indicate different applications of the scheduling scheme. Furthermore, we assume throughout that performance values for different users are *comparable and additive*.

We consider both the uplink and the downlink of a wireless network. In both cases, the base station serves as the scheduling agent. The scheduling scheme does the following: at the beginning of a time slot, the scheduler (i.e., the base station) decides

which user should be assigned the time slot based on the performance values of the users at that time slot. (We describe a particular scheduling procedure in Section III-C, including how the scheduler obtains information about the users' performance values.) For the uplink case, if a user is assigned a time slot, the user will transmit in that time slot. For the downlink case, if a user is assigned a time slot, the base station will transmit to the user in that time slot. If time slot k is assigned to user i, the system is "rewarded" with a performance value of U_i^k, i.e., user i's performance value at time slot k. *The goal of the scheduling scheme is to maximize the average system performance by exploiting the time-varying channel conditions, given the time-fraction assignment*. Basically, the scheduling policy systematically assigns a time slot to a user with a performance value that is large relative to those of the other users, while satisfying the time-fraction requirements of users.

Our scheduling scheme could be implemented in time division multiple access/ frequency division multiple access (TDMA/FDMA) systems as well as time-slotted code division multiple access (CDMA) systems. The length of a time slot in the scheduling policy can be different from an actual time slot of a physical channel. The length of a scheduling time slot depends on how fast the channel conditions vary and how fast we want to track the variation. As mentioned in [12], it is necessary to "track" (at least slow fading) signal-level variations for better network performance.

III. Optimal Scheduling Policy

In this section, we describe the scheduling problem and our scheduling scheme. Let r_i denote the time-fraction assigned to user i, where $\sum_{i=1}^{N} r_i = 1$ and N is the number of users in the cell. Here, we assume that the r_is are predetermined and serve as a prespecified fairness constraint—on average, a fraction r_i of the whole time should be scheduled to user i. Our goal is to develop a scheduling scheme that exploits the time-varying channel conditions to maximize the system performance, under the time-fraction constraints r_i, $i = 1, \ldots, N$.

Let $\vec{U}^k = (U_1^k, \ldots, U_N^k)$ be the *performance vector* at time slot k, where U_i^k is the performance value achieved by user i if time slot k is assigned to user i. We assume that U_i^k is nonnegative and bounded. Assume from now that $\{\vec{U}^k\}$ is stationary, so that the time index k can be dropped. Specifically, we use the notation $\vec{U} = (U_1, \ldots, U_N)$, where U_i is a random variable representing the performance value of user i at a generic time slot.

The scheduling problem is stated as follows: given $\vec{U}$, determine which user should be scheduled (in the given time slot). We define a *policy* Q to be a mapping from the performance-vector space to the index set $\{1, 2, \ldots, N\}$. Given $\vec{U}$, the policy Q determines the user to be scheduled: if $Q(\vec{U}) = i$, then user i should use the time slot, and the system receives a performance "reward" of $U_{Q(\vec{U})}$ (i.e., U_i). Hence, $E(U_{Q(\vec{U})})$ is the average system performance value associated with policy Q. Note that the policy Q is potentially "opportunistic" in the sense that it can use information on the performance vector $\vec{U}$ to decide which user to schedule.

We are interested only in policies that result in satisfaction of the time-fraction assignment constraints. Specifically, we say that a policy Q is *feasible* if $P\{Q(\vec{U}) = i\} = r_i$ for all $i = 1, \ldots, N$. Feasible policies are those that obey the given resource allocation constraints. We use Θ to denote the set of all feasible policies. Our goal is to find a *feasible* policy Q that maximizes the average system performance. The problem can be stated formally as follows:

$$\underset{Q \in \Theta}{\text{maximize}}\, E\Big(U_{Q(\vec{U})}\Big). \tag{1}$$

Note that we can write

$$E\Big(U_{Q(\vec{U})}\Big) = E\left(\sum_{i=1}^{N} U_i \mathbf{1}_{\{Q(\vec{U})=i\}}\right) = \sum_{i=1}^{N} E\Big(U_i \mathbf{1}_{\{Q(\vec{U})=i\}}\Big)$$

where

$$\mathbf{1}_A = \begin{cases} 1, & \text{if } A \text{ occurs} \\ 0, & \text{otherwise} \end{cases}$$

is the indicator function of the event A. In other words, the overall objective function is the sum of all users' average performance values (where we reap a reward of U_i only if user i is scheduled).

Recall that we assumed the sequence $\{\vec{U}^k\}$ to be stationary. This assumption does not preclude correlations across users or across time. In practice, a user's channel condition is usually time-correlated, for example, due to shadowing. Hence, a user's performance is usually also time-correlated. Furthermore, the performance of different users may also be correlated. For example, when the intercell interference is high, some users' performance values simultaneously decrease. However, if users have enough separated locations, it is reasonable to assume that their performance values are only weakly dependent.

In the following, we present our opportunistic scheduling policy. We state the properties of our policy, including its optimality with respect to (1). Then we explain how to estimate the parameters used in the policy. Finally, we describe a procedure to implement our scheduling policy by tuning the parameter values on-line based on measurements.

A. Opportunistic Scheduling Policy

1) Two-User Case: For the purpose of illustration, we start with the two-user case. Suppose that user 1 and user 2 have time-fraction assignments r_1 and r_2, respectively, and $r_1 + r_2 = 1$. We wish to find an opportunistic policy that solves (1).

Define $y(v) = P\{U_1 + v \geq U_2\}$, where $v \in \mathbb{R}$. Because $y(v)$ is the distribution function of the random variable $(U_2 - U_1)$, $y(v)$ is a right-continuous monotonically increasing function of v with $y(\infty) = 1$ and $y(-\infty) = 0$. Hence, there exists a v^* (which may not be unique) such that for any $\epsilon > 0$

$$y(v^* - \epsilon) \leq r_1 \leq y(v^*)$$

where r_1 is the time-fraction assignment of user 1. Let $y^-(v) = P\{U_1 + v > U_2\}$, which is a left-continuous monotonically increasing function of v. So

$$y^-(v^*) \le r_1 \le y(v^*)$$

i.e., $y(v^*) - y^-(v^*) = P\{U_1 + v^* = U_2\} \ge 0$. If $y(v^*) - y^-(v^*) > 0$, then $p = (r_1 - y^-(v^*))/(y(v^*) - y^-(v^*))$. Otherwise, let $p = 1$ (this value does not matter). The opportunistic scheduling policy is then given by

$$Q^*\left(\vec{U}\right) = \begin{cases} 1, & \text{if } U_1 + v^* > U_2 \\ 1, & \text{with prob. } p \text{ if } U_1 + v^* = U_2 \\ 2, & \text{with prob. } 1-p \text{ if } U_1 + v^* = U_2 \\ 2, & \text{if } U_1 + v^* < U_2. \end{cases}$$

It is clear that the policy $Q^*(\vec{U})$ defined above is feasible:

$$P\left\{Q^*\left(\vec{U}\right) = 1\right\} = P\{U_1 + v^* > U_2\} + P\{U_1 + v^* = U_2\}p = r_1.$$

The policy can be described as follows. The space spanned by U_1 and U_2 is divided into two halves by the line $U_1 + v^* = U_2$. Above the line (i.e., $U_2 > U_1 + v^*$), we always schedule user 2 to transmit. Under the line (i.e., $U_1 + v^* > U_2$), we always schedule user 1 to transmit. If the probability of the line is positive, some randomization is needed if we fall on the line—with probability p, we schedule user 1 and with probability $1-p$, we schedule user 2, where $p = (r_1 - y^-(v^*))/(y(v^*) - y^-(v^*))$ is determined by the time-fraction assignment constraint.

2) General Case: Now we extend the policy from the previous section to the N-user case. Define

$$y_i(\vec{v}) = P\left\{U_i + v_i \ge \max_{j \ne i}(U_j + v_j)\right\}, \qquad \text{for } i = 1, \ldots, N$$

where $\vec{v} = (v_1, \ldots, v_N)$. Note that $y_i(\vec{v})$ is a monotonically increasing right-continuous function of v_i and a monotonically decreasing left-continuous function of v_j, $j \ne i$. Hence, there exists a $\vec{v}^*$ that satisfies $P\{Q^*(\vec{U}) = i\} = r_i$, where our opportunistic policy Q^* is defined as

$$Q^*\left(\vec{U}\right) = \arg\max_i (U_i + v_i^*). \tag{2}$$

When ties occur in the argmax above, we break ties probabilistically by picking a user i among those that achieve the maximum above with a certain probability. Note that $\vec{v}^*$ is not unique. There are N components but only $N-1$ independent constraint equations: $P\{Q^*(\vec{U}) = i\} = r_i$, for $i = 1, \ldots, N-1$, and $P\{Q^*(\vec{U}) = N\} = 1 - \sum_{i=1}^{N-1} r_i$ is a linear combination of the first $N-1$ equations. Hence, we can simply set $v_N^* = 0$.

3) Properties: The policy Q^* defined in (2), which represents our opportunistic scheduling policy, is optimal in the following sense.

Proposition 1: The policy Q^* is a solution to the problem defined in (1), i.e., it maximizes the average system performance under the resource allocation constraint.

We can think of the parameter $\vec{v}^*$ in (2) as an "offset" used to satisfy the time-fraction assignment constraint. Under this constraint, the scheduling policy schedules the "relatively-best" user to transmit. User i is "relatively-best" if $U_i + v_i^* \ge U_j + v_j^*$ for all j. In the special case where $v_j^* = 0$ for all j, the scheduling policy reduces to $Q^*(\vec{U}) = \arg\max_i U_i$, i.e., always schedule the user with the largest performance value to transmit. The proof of the optimality is attached in Appendix I.

The policy Q^* maximizes the average system performance even if users' performance values are arbitrarily correlated, both in time and across users. The following proposition establishes, under a more restrictive assumption, that our scheme improves *every user's* average performance relative to any nonopportunistic scheduling policy.

Proposition 2: If the performance values U_is, $i = 1, \ldots, N$, are independent, then

$$E\left(U_i \mathbf{1}_{\{Q^*(\vec{U}) = i\}}\right) \ge r_i E(U_i)$$

where $E(\mathbf{1}_{\{Q^*(\vec{U}) = i\}}) = r_i$.

The proof of this property is attached in Appendix II. Note that $E(U_i \mathbf{1}_{\{Q^*(\vec{U}) = i\}})$ is the average performance of user i when using our opportunistic scheduling policy, and $r_i E(U_i)$ is the average performance of user i when using a nonopportunistic scheduling scheme. This proposition makes a strong statement about the individual performance of each user. If users' performance values are independent, the average performance of *every user* in our opportunistic scheduling scheme will be at least that of any nonopportunistic scheduling scheme. In this sense, the opportunistic scheduling policy does not sacrifice any user's performance to improve the overall system performance. Of course, different users may experience different amounts of improvement. In general, the larger the variance of a user's performance value, the higher the improvement (this observation is corroborated by our experiments; see Section IV).

What Proposition 2 also tells us is that the fraction of time slots assigned to each user is an important measure of the performance of that user. For example, if a user consumes r_i portion of the time slots, then the user is granted a minimum average performance value of $r_i E(U_i)$, when users have independent performance values. (In practice, if users move independently and are not located at the same location, then users usually have independent performance values.)

Moreover, our scheduling scheme can be applied to different scenarios by adopting different forms of performance measures. For example, let us consider the case where each user has a fixed SIR target. By reaching this target SIR, the user gets a certain data rate g_i. Then r_i fraction of the time slots corresponds to a data rate of $r_i g_i$ for user i, assuming that user i always adjusts its power to reach the target SIR. If we define the performance measure as the negative of power consumption used to meet the SIR target, then the scheduling scheme can be applied to minimize the power consumption of users while satisfying certain data rates. Instead, if we define the performance value as the negative of the interference to the other cells, we can minimize the intercell interference while maintaining the data rate.

B. Parameter Estimation

Recall that our opportunistic scheduling policy is given by

$$Q^*\left(\vec{U}\right) = \arg\max_i (U_i + v_i^*)$$

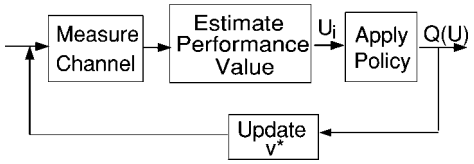

Fig. 2. Block diagram of the scheduling policy with online parameter estimation.

where the v_i^*s are parameters determined by the distribution of $\vec{U}$. In practice, this distribution is unknown, and hence we need to estimate the parameters v_i^*, $i = 1, \ldots, N-1$. Fig. 2 shows a block diagram of a practical scheduling procedure that incorporates online estimation of these parameters.

In this section, we focus on the block that implements the online estimation of the parameters v_i^*, $i = 1, \ldots, N-1$, labeled "Update v^*" in Fig. 2. We first provide an intuitive description of the computation of $\vec{v}^*$ and then present a standard stochastic approximation algorithm to estimate $\vec{v}^*$ (the vector of the v_i^*s).

Let us consider a scheduling algorithm $Q_{\vec{v}}$:

$$Q_{\vec{v}}\left(\vec{U}\right) = \arg\max_i U_i + v_i.$$

If $v_i = 0$ for all i, then, clearly, the scheduling policy maximizes the system performance. If this scheduling policy meets the resource allocation constraint, we can set $\vec{v}^* = \vec{0}$. In general, $\vec{v} = \vec{0}$ does not meet the time-fraction assignment constraint, i.e., there exists at least a user i such that $P\{Q_{\vec{v}}(\vec{U}) = i\} < r_i$. If we increase the value of v_i, we increase the value of $P\{Q_{\vec{v}}(\vec{U}) = i\}$ while decreasing the values of $P\{Q_{\vec{v}}(\vec{U}) = j\}$ for $j \neq i$. The parameters v_i^*s are the offsets that are necessary to satisfy the time-fraction assignment constraint. Intuitively, if $P\{Q_{\vec{v}}(\vec{U}) = i\} < r_i$, we increase the value of v_i, and if $P\{Q_{\vec{v}}(\vec{U}) = i\} > r_i$, we decrease the value of v_i. By making such adjustments, we hope to find the $\vec{v}^*$ eventually such that $P\{Q_{\vec{v}^*}(\vec{U}) = i\} = r_i$ for all i. In the following, we use a standard stochastic approximation algorithm to implement this intuitive idea and to estimate the value of $\vec{v}^*$. The computation involved in the stochastic approximation algorithm is very simple.

We first roughly explain the idea of the stochastic approximation algorithm used in this paper. For a systematic and rigorous study of stochastic approximation algorithms, see [13], [14]. Suppose we want to solve the root-finding problem $f(x^*) = 0$, where f is a continuous function with one root x^* [both x^* and $f(x^*)$ are vectors of the same dimension]. If we can evaluate the value of $f(x)$ at any x, then we can use the iterative algorithm

$$x^{k+1} = x^k - a^k f(x^k)$$

which will converge to x^* as long as the step size a^k is appropriately chosen, e.g., $a^k = 1/k$. Suppose that we cannot obtain exactly the value of $f(x^k)$ at x^k, but instead we only have a noisy observation g^k of $f(x^k)$ at x^k, i.e., $g^k = f(x^k) + e^k$ where e^k is the observation error (noise). In this case, it is well known that if $E(e^k) = 0$ (i.e., the mean of the observation error is zero), then the algorithm

$$x^{k+1} = x^k - a^k g^k$$

converges to x^* with probability 1 under appropriate conditions on a^k and f (see, e.g., [13] and [14]).

In this paper, we use a stochastic approximation algorithm to estimate $\vec{v}^*$. For this, note that we can write $\vec{v}^*$ as a root of the equation $f(\vec{v}^*) = 0$, where the ith component of $f(\vec{v}^*)$ is given by

$$f_i(\vec{v}^*) = P\left\{Q^*\left(\vec{U}\right) = i\right\} - r_i, \qquad i = 1, \ldots, N-1,$$

and

$$Q^*\left(\vec{U}\right) = \arg\max_i (U_i + v_i^*).$$

We use a stochastic approximation algorithm to generate a sequence of iterates $\vec{v}^1, \vec{v}^2, \ldots$ that represent estimates of $\vec{v}^*$. Each $\vec{v}^k$ defines a policy Q^k given by

$$Q^k\left(\vec{U}\right) = \arg\max_i \left(U_i + v_i^k\right).$$

To construct the stochastic approximation algorithm, we need an estimate g^k of $f(\vec{v}^k)$. Note that, although we cannot obtain $f(\vec{v}^k)$ directly, we have a noisy observation of its components

$$g_i^k = \mathbf{1}_{\{Q^k(\vec{U})=i\}} - r_i, \qquad i = 1, \ldots, N-1.$$

The observation error in this case is

$$e_i^k = g_i^k - f_i(\vec{v}^k) = \mathbf{1}_{\{Q^k(\vec{U})=i\}} - P\left\{Q^k\left(\vec{U}\right) = i\right\}$$

and thus we have $E(e_i^k) = 0$. Hence, we can use a stochastic approximation algorithm of the form

$$v_i^{k+1} = v_i^k - a^k\left(\mathbf{1}_{\{Q^k(\vec{U})=i\}} - r_i\right)$$

where, e.g., $a^k = 1/k$. For the initial condition, we can set v_i^1 to be 0, or some estimate based on the measurement history. For the above algorithm, following the standard proof in [13], we can show that $\{v_i^k\}$ converges to v_i^* with probability 1. Furthermore, to accelerate the convergence and to reduce the range of the fluctuation of the stochastic approximation algorithm, we can use the standard technique of averaging (see, e.g., [14])

$$\overline{v}_i^k = \left(1 - \frac{1}{k}\right)\overline{v}_i^{k-1} + \frac{1}{k}\, v_i^k.$$

Our simulations show that, with the stochastic approximation algorithm, v_i^k converges to v_i^* relatively quickly.

When the U_is are not continuous random variables, there may be "ties" in the argmax of Q^* in (2). Specifically, ties occur when $P\{U_i + v_i^* = \max_{j \neq i}(U_j + v_j^*)\} > 0$ for some i. In this case, v_i^k will still converge to v_i^*. However, we should break ties probabilistically by picking a user i among those that achieve the maximum with a certain probability. In practice, we do not need to estimate this probability because the tiebreak can be handled automatically by the adaptive nature of the stochastic approximation algorithm. To see this, imagine v_i^k fluctuating around v_i^* within a small range; when v_i^k is too large, we have $P\{Q = i\} > r_i$, and hence v_i^k will be dragged down. Our simulation results show that the stochastic approximation algorithm works well in both the continuous and "tie-break" cases—the system performance obtained with the stochastic approximation scheme is very close to that of the true optimal value while maintaining the resource allocation requirements.

C. Implementation Considerations

So far, we have described our optimal scheduling policy and addressed the problem of estimating the parameter values

needed for the policy. In this section, we explore some implementation considerations for our scheduling policy.

In our scheduling policy, the base station needs to obtain information of each user's performance value at a given time slot to make the scheduling decision. The performance value of a user can be estimated either by the user or by the base station, based on the channel condition and/or measurements from previous transmissions. For the downlink case, a user could measure the received signal power level (from the user's base station) and the interference power level. The user could then calculate the performance value of the time slot based on the channel condition and other factors (such as power consumption). For example, suppose a user's performance measure is its throughput, which is a linear function of the SINR, as shown in Fig. 1. Based on the estimated SINR, the user can then obtain its performance value. For the uplink case, the base station could estimate the user's channel condition based on the received signal from the user. Assuming the base station knows the form of the performance value for each user (i.e., how the performance value depends on the SINR and/or other factors), the performance value could then be calculated by the base station.

If the performance value is estimated by the user, this information needs to be sent to the base station, which can be accomplished in several ways. For example, each user could maintain a small signaling channel with the base station. Alternatively, the required information could be piggybacked over the user's acknowledgment packets.

As mentioned before, the length of a time slot in our scheduling policy can be different from an actual time slot of the physical channel. The length of a scheduling time slot depends on how fast the channel condition varies and how fast we want to track the variation. The usual tradeoff between accuracy and signaling overhead exists here. Specifically, more frequent updating provides more accurate tracking of varying channel conditions, but incurs higher signaling costs. In practice, to decrease signaling costs, a user can update its information only when the change in the performance value is larger than a certain threshold. Furthermore, it is not necessary for all users to update at the same time. Note that propagation delay is ignored.

In the following, we summarize our scheduling procedure, which incorporates the online parameter estimation algorithm described in the last section. As mentioned before, the initial estimate $\vec{v}^1$ can be set to $\vec{0}$ or some value based on history information. At each time slot $k = 1, 2, \ldots$, the system performs the following steps.

1) Estimate U_i^k.
   - Uplink: the base station estimates each user's channel condition and calculates the values of U_i^k, $i = 1, \ldots, N$.
   - Downlink: user i measures its channel condition, calculates U_i^k, and informs the base station.
2) The base station decides which user should be scheduled to transmit in the time slot based on the scheduling policy:

$$Q^k\left(\vec{U}^k\right) = \arg\max_i \left(U_i^k + v_i^k\right).$$

3) The scheduled transmission takes place.
   - Uplink: the base station broadcasts the ID of the selected user, and the selected user transmits in the time slot.
   - Downlink: the base station transmits to the selected user.
4) The base station updates the parameter vector $\vec{v}^{k+1}$ via

$$v_i^{k+1} = v_i^k - a^k\left(\mathbf{1}_{\{Q^k(\vec{U}^k)=i\}} - r_i\right).$$

For the stationary case, we set $a^k = 1/k$. For the nonstationary case, we set a^k to a small constant to track system variations.

Note that the computation burden above is $O(N)$ per time slot, where N is the number of users sharing the channel (usually on the order of tens), which suggests that the procedure is easy to implement in practice.

IV. Simulation Results

In this section, we present numerical results from computer simulations of our scheduling scheme. Our scheduling policy exploits time-varying channel conditions—the policy dynamically decides which user should be scheduled to transmit in a time slot based on users' current performance values. For the purpose of simulations, we assume that the time-fraction assignment is done using *fair sharing*, i.e., the total resources are evenly divided among the users. The well known *round-robin* scheme is a policy that shares the resource (time in this case) in this manner, but does not exploit channel conditions. To evaluate the performance gain of our dynamic and opportunistic assignment of transmissions, we compare the performance of our policy with that of the round-robin scheme. We will show two sets of simulation results. The first one is to simulate a cellular system using our scheduling policy and evaluate the improvement of our scheme. Then, we show how estimation errors on performance values affect the results of the scheduling scheme, and how the stochastic approximation works.

A. Cellular Model

Our simulation environment is described in the following. We consider a multicell system consisting of a center hexagonal cell surrounded by hexagonal cells of the same size. The base station is at the center of each cell, and simple omnidirectional antennas are used by mobiles and base stations. The frequency reuse factor is 3, and the co-channel interferences from the six first-ring neighboring cells are taken into account. We assume that each cell has a fixed number of frequency bands, and we focus on one frequency band in the central cell, which is shared by 25 users. The scheduling policy decides which user should transmit in this frequency band at each time slot.

Users move with random speed and direction in the cell and have exponentially distributed "on" and "off" periods. They perceive time-varying and location-dependent channel gains. The channel gains of the users are mutually independent random processes determined by the sum of two terms: one due to path (distance) loss and the other to shadowing. We adopt the path-loss model (Lee's model) and the slow log-normal

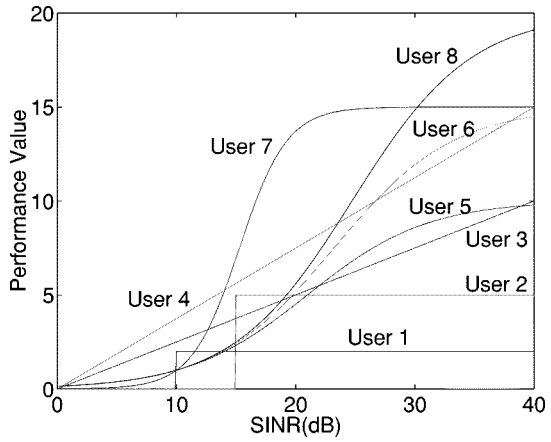

Fig. 3. Users' performance values as a function of SINR.

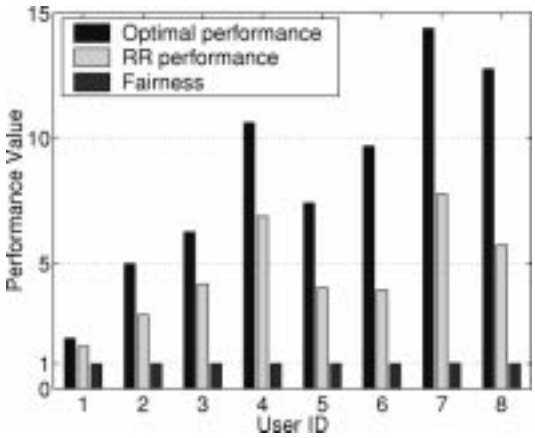

Fig. 4. Comparison of the opportunistic scheduling policy with the round-robin scheme.

shadowing model in [15]. To be conservative, we ignore the effects of fast multipath fading in the simulation. If fast fading could be tracked accurately, our scheme would provide even higher performance improvements than shown here. The mobility model, the propagation model, and the parameters of the simulation are discussed in detail in [5].

Fig. 3 shows the forms of the performance values used by different users. To avoid crowding the figure, we only show the performance functions of eight users. The performance values of users 1 and 2 are step-functions of their SINR, and user 2 has a higher threshold than user 1. The performance values of users 3–4 are linear functions of their SINR (in dB), with different slopes. Users 5–8 have performance values that are S-shape functions of their SINR, with different parameters. Totally, there are four users with step-functions, six users with linear functions, and 15 users with S-shape functions in the simulation.

As mentioned earlier, for our simulation experiments, we assume *fair sharing* time-fraction assignment. When the number of active users N changes, i.e., when an active user becomes inactive or vice versa, we update the time-fraction assignment r_i for all active users. In other words, if N is the number of active users sharing the channel in the central cell, we set $r_i = 1/N$, where user i is active.

In each time slot, the system performs the following: all active users inform the base station their estimated performance values; the base station decides which user to transmit using the opportunistic scheduling policy, and updates the parameters; and then the selected user transmits. For a detailed simulation procedure, we refer interested readers to [5].

Fig. 4 shows the results of our simulation experiment. In the figure, the x axis represents the users' IDs. For each user, we compare the average performance in our opportunistic scheduling policy (the first bar) with that of the round-robin policy (the second bar). We can see that, in every case, our opportunistic policy significantly outperforms the nonopportunistic round-robin policy, with gains of 20% to 150%. The amount of improvement varies from user to user because different users have different performance functions. The third (right-most) bar in the figure is the ratio of the total number of slots assigned to each user in our opportunistic scheme to that of the round-robin scheme, which is equal to that required by the time-fraction assignment constraint. For all users, the third bar is virtually identical to 1. Hence, our scheduling scheme satisfies the time-fraction assignment constraint, which suggests that our stochastic approximation algorithm works well in the simulation experiment even in the nonstationary case.

TABLE I
GAUSSIAN PROCESS PARAMETERS

ID	mean	stand. deviation	corr. coefficient
user 1	10	10.8	0.3
user 2	10	6.9	0.4
user 3	10	4.0	0.5
user 4	10	2.5	0.6
err. 1 (e_i^1)	0	4	0
err. 2 (e_i^2)	0	8	0

B. Estimation Errors

When our scheduling policy is implemented, the following errors may occur. The first is the estimation error on $\vec{v}^*$ using the stochastic approximation algorithm, i.e., the discrepancy between $\vec{v}^k$ and $\vec{v}^*$. Second, imperfect measurement of the channel conditions may introduce errors in the estimates of users' performance values. This experiment is designed to evaluate the impact of our online parameter estimation procedure and the sensitivity of our opportunistic scheduling scheme to estimation errors on users' performance values. We generate four time-correlated Gaussian processes, representing the performance-value sequences for four users. The means and standard deviations of the four Gaussian processes are displayed in Table I. Each user has exponentially distributed "on" and "off" periods, which are used to generate status changes in order to test how well the stochastic approximation tracks the changes. We compare four different cases:

- ideal case, i.e., Q^* with known threshold $\vec{v}^*$ assuming exact values of U_is are known;
- estimated thresholds using the stochastic approximation algorithm assuming exact values of U_is are known;

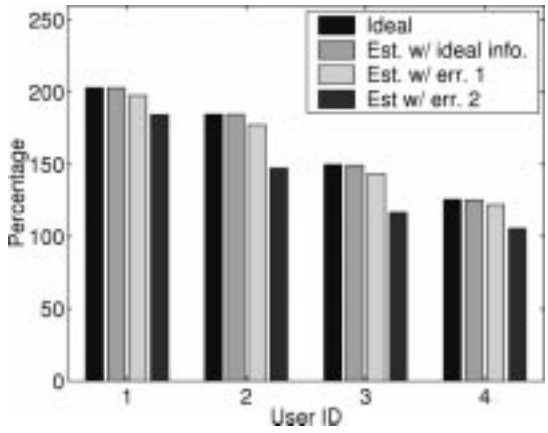

Fig. 5. Average performance value, normalized over the round-robin scheme.

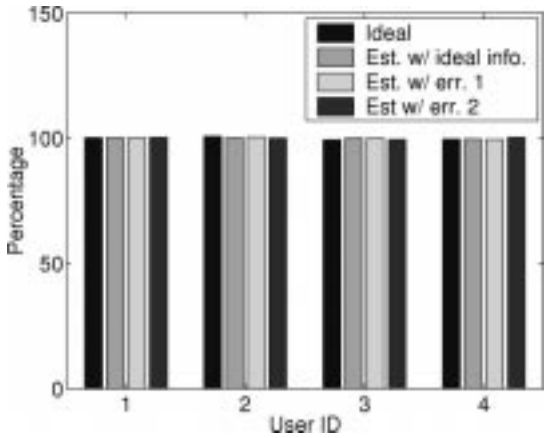

Fig. 6. Fairness, normalized over the round-robin scheme.

- estimated thresholds with estimated value of U_i; i.e., $\hat{U}_i = U_i + e_i^1$, where e_i^1 is the estimation error on user i's performance value, $e_i^1 \sim N(0, 4)$ (as in Table I), and the e_i^1s are independent;
- estimated thresholds with estimated value of U_i, i.e., $\hat{U}_i = U_i + e_i^2$, where e_i^2 is the estimation error on user i's performance value, $e_i^2 \sim N$ (0, 8) (as in Table I), and the e_i^2s are independent.

Fig. 5 shows the average performance (normalized over the average performance value of the round-robin scheme) from the above four cases. The first bar is the normalized performance value under the ideal condition; the second bar is that of estimated thresholds with ideal measurements (i.e., the exact value of U_i is known). We can see that the performance of both the optimal policy Q^* and our online policy Q^k with estimated parameters are quite comparable and are significantly higher than that of the round-robin policy. This indicates that our online parameter estimation scheme works well and that errors due to the parameter estimation do not significantly degrade the performance of the policy relative to the optimal policy.

In Fig. 5, the performance gains appear to be related to the standard deviation: the higher the standard deviation, the larger the performance gain. Note that, in the round-robin scheme, the performance levels for all users are all approximately equal to the mean of the Gaussian processes (which is the mean performance value). This is to be expected because the round-robin scheme allocates an equal fraction of time slots to each user, regardless of the channel conditions. Our opportunistic approach takes advantage of favorable transmission conditions, thereby leading to average performance values that are far above the mean of the Gaussian processes.

The third and fourth bars in Fig. 5 represent the normalized performance with different estimation errors. The third bar shows the result of the case with estimation error $e_i \sim N$ (0, 4). With this estimation error, the average performance is still close to that of the optimal case. When the estimation error increases, it is not surprising that the average performance decreases. The fourth bar shows a situation with very large estimation errors, especially for user 4. However, even in this case, our scheduling scheme still outperforms that of the round-robin. This suggests that our opportunistic scheduling scheme is robust to estimation errors on users' performance values.

Fig. 6 shows the ratio of the time fractions obtained by our policy Q^k to that of the round-robin policy (which, as pointed out before, are equal to the prespecified values). As we can see, our scheme satisfies the time-fraction constraints very well in all four cases, even in the case with very large estimation errors.

V. Short-Term Performance

The scheduling scheme described thus far meets the long-term performance requirements of users, i.e., the long-term average of the fraction of time slots assigned to a user is guaranteed. However, with such a scheme, it is possible that a user could be starved for a long time (say, more than a few seconds), which may be undesirable for certain users. Usually, a user may also have the demand for good "short-term" performance—the user expects that the amount of service obtained within a relatively short time interval be close to the amount it should get.

In the generalized processor sharing (GPS) model [16], each flow is treated as a fluid flow. Each flow i is given a weight w_i, and for any time interval $[t_1, t_2]$ during which both sessions i and j are continuously backlogged, the resource granted to each flow i, $G_i(t_1, t_2)$, satisfies the following property:

$$\left| \frac{G_i(t_1, t_2)}{w_i} - \frac{G_j(t_1, t_2)}{w_j} \right| = 0. \tag{3}$$

This means that a user gets its fair share of resource during any time interval. There is an alternative to (3). Let t_0 be a starting point such that from time t_0 onwards, both sessions i and j are continuously backlogged. It is clear that the satisfaction of (3) is equivalent to

$$\left| \frac{G_i(t_0, t)}{w_i} - \frac{G_j(t_0, t)}{w_j} \right| = 0 \tag{4}$$

for all $t > t_0$, and users i and j are both continuously backlogged during the time interval $[t_0, t]$.

We extend the above concept to a time-slotted system, where one time slot is exclusively used by one user. Let w_i be the weight of user i and $r_i(k)$ be the time-fraction requirement of user i at time k. Note that, when the set of active users change,

$r_i(k)$s may change. For example, following the tradition in GPS, $r_i(k)$ can be set as

$$r_i(k) = \frac{w_i}{\sum_{j \in A_k} w_j}$$

where A_k is the set of active users at time k. User i is guaranteed a minimum share of the resource $w_i / \sum_{j \in A} w_j$ during its active period, where A is the set of all users.

Let k_i be the time that user i becomes active. Suppose that, during the time interval $[k_0, k]$, both users i and j are continuously active, where $k_0 = \max(k_i, k_j)$. Let $S_i(k_0, k)$ be the number of time slots assigned to user i from k_0 to k. An approximation of (4) is

$$\left| \frac{S_i(k_0, k)}{w_i} - \frac{S_j(k_0, k)}{w_j} \right| \leq \Gamma$$

where $\Gamma \geq 0$ is a constant.

Let $F_i(k_i, k)$ be the counter of the resource entitled of user i, i.e., the number of time slots that should be assigned to user i during the time interval $[k_i, k]$

$$F_i(k_i, k) = \sum_{t=k_i}^{k} r_i(t) = \sum_{t=k_i}^{k} \frac{w_i}{\sum_{j \in A_t} w_j} \tag{5}$$

where A_t is the set of active users at time t. It is obvious that the $F_i(k_i, k)$s satisfy (4) and hence (3) at each discrete time k. Note that $F_i(k_i, k)$ may not be an integer and thus may not be achievable when a time slot is exclusively used by one user.

We use $F_i(k_i, k)$ as a benchmark. To improve the short-term performance, we want $S_i(k_i, k)$ to be close to $F_i(k_i, k)$. We modify our previous opportunistic scheduling scheme in the following way. Let

$$\Delta_i^k = F_i(k_i, k) - S_i(k_i, k).$$

If $\Delta_i^k > 0$, then user i is "lagging" (i.e., the user gets less resource than it should get), and if $\Delta_i^k < 0$, then user i is "leading." The idea is to increase the probability of transmission of a lagging user and decrease the probability of transmission of a leading user. Hence, a direct modification of our scheduling policy is the policy B^k given by

$$B^k\left(\vec{U}^k\right) = \arg\max_{i=1,\ldots,N} (U_i^k + v_i^*)\left(\frac{\Delta_i^k}{\alpha} + \beta\right) \tag{6}$$

where α and β are positive constants. When the value of α is smaller, the effect of Δ_i^k is more significant, and thus the short-term performance is better. The value of β acts as a threshold—a user is forbidden to transmit if the amount by which it leads is greater than $\beta\alpha$.

Next, we consider the case where there are changes in the set of active users. When a new user comes into the system, the system adjusts the r_is for all users, and the new user j starts a counter $F_j(k_j, k)$ for its resource share. When a user leaves the system, the system adjusts the r_is and the counters of fair share for other active users. Suppose user i leaves the system at time k and user i has been served with $S_i(k_i, k)$ time slots. Recall that $\Delta_i^k = F_i(k_i, k) - S_i(k_i, k)$, i.e., user i has Δ_i^k time slots less than its share. Because the user is gone, we cannot force $S_i(k_i, k)$ to be close to $F_i(k_i, k)$ anymore. This discrepancy has to be absorbed by other active users. We update the counter $F_j(k_j, k)$ of any active user j by replacing the value of $F_j(k_j, k)$ by $F_j(k_j, k) + \Delta_i^k / N_{act}$, where N_{act} is the number of active users. In other words, the discrepancy is evenly distributed among all active users. Note that this way of handling users' departures is intuitive, but not necessarily optimal. Actually, it is challenging to even define a good optimal criterion in the situation where there exists the tradeoff between short-term performance and the overall system performance.

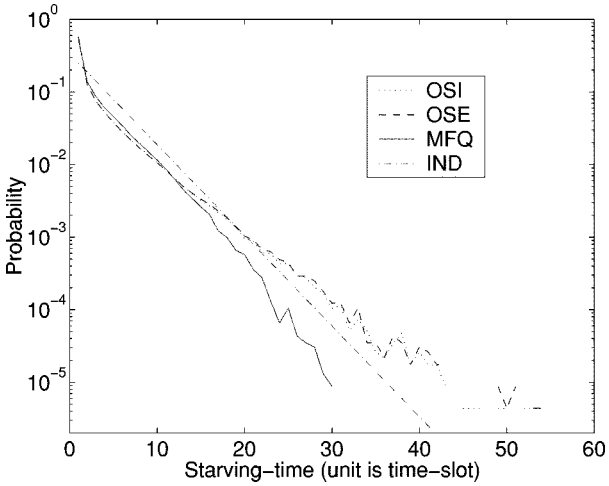

Fig. 7. Starving-time histogram.

We use the same simulation setup as in Section IV-B. Four Gaussian random processes are used to represent the performance-value sequences of four users, and their parameters are shown in Table I. The simulation runs for 1 000 000 time slots (while all four users have exponentially distributed on–offs). Next, we show two metrics for the short-term performance for user 4, which has a time-correlation coefficient of 0.6.

The first metric is the starving-time, defined as the time interval between two contiguous time-slot assignments when the user is active. Note that starving-time is closely related to the delay a user experiences. Fig. 7 shows the starving-time histogram. In the legend, MOS represents the modified opportunistic scheduler defined in (6); OSI is the ideal opportunistic scheduler with known threshold $\vec{v}^*$; OSE is the opportunistic scheduler using stochastic approximation to estimate the threshold; and IND represents the numerical result when a user's performance values at different time slots are independent. If a user's performance values are independent across time, the starving-time is binomially distributed, i.e., $P\{\text{starving-time} = n\} = (1 - r_i)^{(n-1)} r_i$, where r_i is the time-fraction of user i. Because user 4's performance value is correlated across time, compared with the IND case, the probability of a long starving-time of user 4 in the OSI and OSE cases, which do not consider the short-term performance, is larger. Furthermore, the probability that a large starving-time occurs in MOS is much smaller than that of OSI and OSE, and it is also smaller than that of IND. Hence, the chance that a user is starved decreases and the short-term performance is improved.

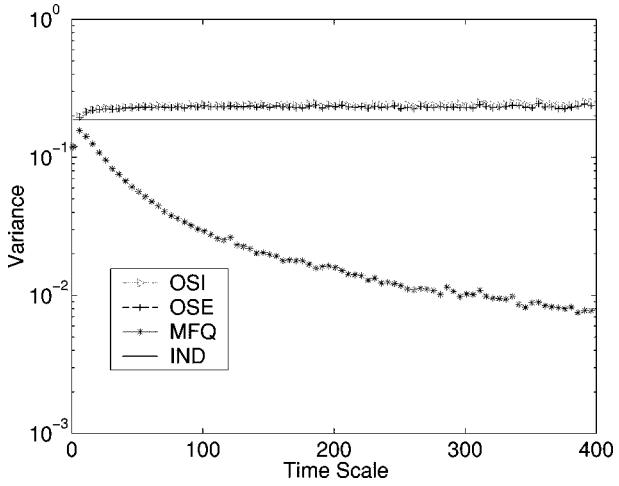

Fig. 8. Normalized variance of discrepancy as a metric of the short-term performance.

The second metric of the short-term performance is defined as

$$\frac{E\Big[\big(S_i(k,\, k+\Delta m) - F_i(k,\, k+\Delta m)\big)^2\Big]}{\Delta m}$$

where Δm is the length of the window by which we measure the discrepancy between the fair share $F_i(k,\, k+\Delta m)$ and $S_i(k,\, k+\Delta m)$ while user i is active during the interval $[k,\, k+\Delta m]$. Because $E[S_i(k,\, k+\Delta m)] = F_i(k,\, k+\Delta m)$, this metric is the variance of $S_i(k,\, k+\Delta m)$ normalized over the window size Δm. The discrepancy should be zero in the Fluid Fair Model and is no larger than 1 in the round-robin scheduling scheme. In Fig. 8, we show that MOS results in a noticeable decrease of the normalized variance.

We should mention here that the modified scheduling scheme does not decrease the average performance significantly. In this simulation, OSI outperforms round-robin by 25% in terms of the average performance of user 4, and MOS outperforms round-robin by 22% while satisfying the long-term resource allocation requirement. Overall, the system performance obtained in MOS is only about 3% less than that of OSI, while both outperforming the round-robin by over 60% and satisfying the long-term resource allocation requirement. Hence, MOS improves the short-term performance without dramatically decreasing the system throughput. In general, the larger the time-correlation, the worse the short-term performance, and the greater the improvement in the short-term performance, the larger the loss in system performance.

One closely related problem is to be able to handle users with explicit delay requirements, such as audio and video. It is a challenging problem to schedule users opportunistically while satisfying the delay requirements of certain users. One possible solution is to extend the current scheduling scheme in the following way. Each user has a due time known by the base station. The due time of a user is the due time of the first packet in the user's queue. If a user has no delay requirement, its due time is set to be ∞. Suppose different users have different due-times. (If two users have the same due-time, we randomly pick one and make its due time the time slot before its actual due time.) We then adjust the transmission probability of a user of according to its due time. The closer the due time, the higher the probability of its transmission. When a user is at its due time, we assign the time slot to this user with probability 1. If we know the distribution functions of users' performance values, it is possible to determine how large the transmission probability should be (as a function of its and other users' due times) numerically. Otherwise, these transmission probabilities might be determined experimentally.

VI. Conclusion

In this paper, we present an opportunistic transmission-scheduling policy. Given a time-fraction assignment requirement, the scheduling policy maximizes the average system performance. In our model, each user's performance value is a stochastic process, reflecting the time-varying performance that results from randomly-varying channel conditions. The users' performance-value processes can be arbitrarily correlated, both in time and across users. We establish the optimality of our opportunistic scheduling policy. We also provide a scheduling procedure that includes an on-line parameter-estimation algorithm to estimate parameter values used in the scheduling policy. Our scheduling algorithm has a low computational burden, which is important for on-line implementation. Via simulation, we illustrate the performance of our scheduling policy, showing significant performance gains over the round-robin policy. Our simulation results also show that our scheme works well for the case of nonstationary performance-value sequences and is robust to estimation errors.

Resource allocation and scheduling schemes are important in wireless networks, especially to provide high-rate data and seamless service for future wireless networks. There are many interesting problems in this area that remain to be resolved. These include the need for a general fairness criterion tailored to wireless networks and dealing with the short-term performance or explicit delay requirement for certain users.

Appendix I
Optimality of the Opportunistic Scheduling Policy

Recall that there exists a $\vec{v}^*$ that satisfies $P\{Q^*(\vec{U}) = i\} = r_i$, where the policy Q^* is defined as

$$Q^*\Big(\vec{U}\Big) = \arg\max_i (U_i + v_i^*).$$

In the following, we show that Q^* defined above is an optimal policy, i.e., that $E(U_{Q(\vec{U})}) \le E(U_{Q^*(\vec{U})})$ for any Q satisfying $P\{Q(\vec{U}) = i\} = r_i$.

Let Q be a policy satisfying $P\{Q(\vec{U}) = i\} = r_i$ for all i. Then,

$$\begin{aligned} E\Big(U_{Q(\vec{U})}\Big) &= E\Big(U_{Q(\vec{U})}\Big) + \sum_{i=1}^{N} v_i^*\Big(P\Big\{Q\Big(\vec{U}\Big) = i\Big\} - r_i\Big) \\ &= E\left(\sum_{i=1}^{N} U_i \mathbf{1}_{\{Q(\vec{U})=i\}}\right) \\ &\quad + \sum_{i=1}^{N} v_i^*\Big(P\Big\{Q\Big(\vec{U}\Big) = i\Big\} - r_i\Big) \\ &= E\left(\sum_{i=1}^{N} (U_i + v_i^*)\mathbf{1}_{\{Q(\vec{U})=i\}}\right) - \sum_{i=1}^{N} v_i^* \tau_i. \end{aligned}$$

By the definition of Q^*, we have

$$E\left(\sum_{i=1}^{N}(U_i+v_i^*)\mathbf{1}_{\{Q(\vec{U})=i\}}\right) \le E\left(\sum_{i=1}^{N}(U_i+v_i^*)\mathbf{1}_{\{Q(^*\vec{U})=i\}}\right).$$

Hence

$$\begin{aligned}E\left(U_{Q(\vec{U})}\right) &\le E\left(\sum_{i=1}^{N}\left(U_i+v_i^*\right)\mathbf{1}_{\{Q(^*\vec{U})=i\}}\right)-\sum_{i=1}^{N}v_i^*\tau_i\\ &= E\left(U_{Q^*(\vec{U})}\right)+\sum_{i=1}^{N}v_i^*\left(P\left\{Q^*\left(\vec{U}\right)=i\right\}-r_i\right)\\ &= E\left(U_{Q^*(\vec{U})}\right)\end{aligned}$$

which completes the proof. □

Appendix II
Performance Improvement for Individual Users

Let T_i be the average performance of user i in the opportunistic scheduling policy, i.e., $T_i = E(U_i\mathbf{1}_{\{Q(\vec{U})=i\}})$. Let T_i^r be the average performance of a nonopportunistic scheduling policy, i.e., $T_i^r = r_iE(U_i)$. In the following, we will show that $T_i \ge T_i^r$ for all i if users have independent performance values. Note that we can write

$$\begin{aligned}T_i &= E\left(U_i\mathbf{1}_{\{Q(\vec{U})=i\}}\right)\\ &= E_{U_i}\left(U_i|Q\left(\vec{U}\right)=i\right)P\left\{Q\left(\vec{U}\right)=i\right\}\\ &= r_iE_{U_i}\left(U_i|Q\left(\vec{U}\right)=i\right).\end{aligned}$$

Hence, to prove that $T_i \ge T_i^r$, it suffices to prove that

$$E_{U_i}\left(U_i|Q\left(\vec{U}\right)=i\right) \ge E_{U_i}(U_i).$$

We have

$$\begin{aligned}Q\left(\vec{U}\right)=i &\Rightarrow U_i+v_i^* \ge U_j+v_j^*, \qquad \text{for all } j\\ &\Leftrightarrow U_i \ge \max_{j\ne i}(U_j+v_j^*)-v_i^*.\end{aligned}$$

Letting $Y = \max_{j\ne i}(U_j+v_j^*)-v_i^*$, we have

$$\begin{aligned}&E_{U_i}\left(U_i|Q\left(\vec{U}\right)=i\right)\\ &\quad\ge E_{U_i}\left(U_i|U_i\ge Y\right)\\ &\quad= E_{U_i}\left(U_i|U_i\ge Y\right)P(U_i\ge Y)\\ &\qquad+E_{U_i}\left(U_i|U_i<Y\right)P(U_i<Y)\\ &\quad\ge E_{U_i}\left(U_i|U_i\ge Y\right)P(U_i\ge Y)\\ &\qquad+E_{U_i}\left(U_i|U_i<Y\right)P(U_i<Y)\\ &\quad= E_{U_i}(U_i),\end{aligned}$$

where $E_{U_i}(U_i|U_i \ge Y) \ge Y \ge E_{U_i}(U_i|U_i < Y)$ due to the hypothesis that U_i is independent of Y. Hence, we have $E_{U_i}(U_i|Q(\vec{U})=i) \ge E_{U_i}(U_i)$, and thus $T_i \ge T_i^r$ for all i. □

References

[1] H. Zhang, "Service disciplines for guaranteed performance service in packet-switching networks," *Proc. IEEE*, vol. 83, pp. 1374–1396, Oct. 1995.

[2] S. Nanda, K. Balachandran, and S. Kumar, "Adaptation techniques in wireless packet data services," *IEEE Commun. Mag.*, vol. 38, pp. 54–64, Jan. 2000.

[3] H. Varian, "Equity, envy, and efficiency," *J. Economic Theory*, vol. 9, pp. 63–91, 1974.

[4] R. Mazumdar, L. Mason, and C. Douligeris, "Fairness in network optimal flow control: Optimality of product forms," *IEEE Trans. Commun.*, vol. 39, pp. 775–782, May 1991.

[5] X. Liu, E. K. P. Chong, and N. B. Shroff, "Transmission scheduling for efficient wireless network utilization," in *IEEE INFOCOM'01*, vol. 2, Alaska, April 2001, pp. 776–785.

[6] S. Lu, V. Bharghavan, and R. Srikant, "Fair scheduling in wireless packet networks," *IEEE/ACM Trans. Networking*, vol. 7, pp. 473–489, Aug. 1999.

[7] T. Nandagopal, S. Lu, and V. Bharghavan, "A unified architecture for the design and evaluation of wireless fair queueing algorithms," in *ACM Mobicom '99*, August 1999.

[8] T. Ng, I. Stoica, and H. Zhang, "Packet fair queueing algorithms for wireless networks with location-dependent errors," in *IEEE INFOCOM'98*, vol. 3, New York, 1998, pp. 1108–1111.

[9] P. Bender, P. Black, M. Grob, R. Padovani, N. Sindhushayana, and A. Viterbi, "CDMA/HDR: A bandwidth-efficient high-speed wireless data service for nomadic users," *IEEE Commun. Mag.*, pp. 70–77, July 2000.

[10] A. Jalali, R. Padovani, and R. Pankaj, "Data throughput of CDMA-HDR a high efficiency-high data rate personal communication wireless system," in *IEEE Veh. Technol. Conf.*, 2000.

[11] F. Kelly, "Charging and rate control for elastic traffic," *Eur. Trans. Telecommun.*, vol. 8, pp. 33–37, 1997.

[12] J. Zander, "Trends in resource management future wireless networks," in *Future Telecommunications Forum, FTF 99*, Beijing, China, Dec. 1999.

[13] H. Kushner and G. Yin, *Stochastic Approximation Algorithms and Applications*. New York: Springer-Verlag, 1997.

[14] I. J. Wang, E. K. P. Chong, and S. R. Kulkarni, "Weighted averaging and stochastic approximation," *Math. Contr. Signals Syst.*, vol. 1, no. 10, pp. 41–60, 1997.

[15] G. Stuber, *Principles of Mobile Communication*. Norwell, MA: Kluwer, 1996.

[16] A. Parekh and R. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: the single-node case," *IEEE/ACM Trans. Networking*, vol. 1, pp. 344–357, May/June 1993.

Xin Liu received the B.S. and M.S. degrees in electrical engineering from Xi'an Jiaotong University, People's Republic of China, in 1994 and 1997, respectively. She is currently working toward the Ph.D. degree in the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN.

She is a Research Assistant in the Network Engineering and Wireless Systems Group at Purdue. Her research interest include wireless networks, resource allocation and scheduling, and QoS provisioning.

Edwin K. P. Chong received the B.E.(Hons.) degree with First Class Honors from the University of Adelaide, South Australia, in 1987 and the M.A. and Ph.D. degrees from Princeton University, in 1989 and 1991, respectively.

While at Princeton, he held an IBM Fellowship. He joined the School of Electrical and Computer Engineering at Purdue University, West Lafayette, IN, in 1991, where he was named a University Faculty Scholar in 1999, and was promoted to Professor in 2001. Since August 2001, he has been a Professor of Electrical and Computer Engineering at Colorado State University, Fort Collins. His current interests are in communication networks and optimization methods. He coauthored the recent book, *An Introduction to Optimization* (New York: Wiley-Interscience, 2001, 2nd ed.).

Dr. Chong received the NSF CAREER Award in 1995 and the ASEE Frederick Emmons Terman Award in 1998.

Ness B. Shroff received the B.S. degree from the University of Southern California, the M.S.E. degree from the University of Pennsylvania, Philadelphia, and the M.Phil and Ph.D. degrees from Columbia University, New York, NY.

He is currently an Associate Professor in the School of Electrical and Computer Engineering at Purdue University, West Lafayette, IN. While completing his doctoral studies, he worked at AT&T Bell Labs (1991) and Bell Communications Research (1992), on problems involving fault management in telephone networks. His current research interests are in wireline and wireless communication networks. He is especially interested in studying issues related to performance modeling and analysis, routing, network management, scheduling, and control in such networks. He also works on problems related to source coding, vector quantization, and error concealment.

Dr. Shroff has received research and equipment grants from the National Science Foundation, AT&T, Hewlett Packard, Intel, LG Electronics, Nortel Networks, Indiana Department of Transportation, Indiana 21st Century program, and the Purdue Research Foundation. He received the NSF CAREER award from the National Science Foundation in 1996. He has served on the technical program committees of various conferences and on NSF review panels. He was the Conference Chair for the 14th Annual IEEE Computer Communications Workshop (CCW) and the Program Co-Chair for the High-Speed Networking Symposium at Globecomm 2000. He is currently on the editorial board of the IEEE/ACM TRANSACTIONS ON NETWORKING and the Computer Networks journal. He is a past editor of IEEE COMMUNICATION LETTERS.