Transmission Scheduling for Efficient Wireless Resource Utilization with Minimum-Performance Guarantees

Xin Liu, Edwin K. P. Chong, and Ness B. Shroff
School of Electrical and Computer Engineering
Purdue University
West Lafayette, IN 47907-1285
{xinliu, echong, shroff}@ecn.purdue.edu

Abstract— **We present an "opportunistic" transmission scheduling scheme that exploits time-varying channel conditions and maximizes the average system performance under minimum-performance guarantees. We establish the optimality of the scheduling scheme, and show that the proposed opportunistic scheduling scheme can provide a "no-loss" guarantee compared to non-opportunistic scheduling policies. Furthermore, we show that the feasibility region of users' requirements is convex, and discuss the associated admission control issues. Last, through simulation results, we show that the scheme results in significant performance improvement.**

Keywords— **Scheduling, wireless, time-varying channel, minimum-performance guarantee, time-slotted system.**

I. Introduction

WIRELESS frequency is a scarce resource, hence efficient spectrum utilization is very important, especially for wide-band data communications. Resource allocation schemes and scheduling policies are critical to achieving this goal. In wireless networks, the channel conditions of mobile users are time-varying because of radio propagation phenomena, such as path-loss variation with distance, slow log-normal shadowing, and fast multi-path fading. Due to variations in channel conditions, users perceive time-varying service quality and/or quantity. Hence, a good scheduling scheme should be able to exploit the variation in channel conditions to improve the spectrum efficiency. 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 the level of satisfaction among different users. This motivates our work: to improve wireless resource efficiency by exploiting time-varying channel conditions, while at the same time guaranteeing the minimum-performance requirement of each user. There are some research efforts on the area of opportunistic scheduling schemes, such as [1], [2], [3], [4].

[1]Beginning August 2001, Edwin K. P. Chong will be with the Dept. of Electrical and Computer Engineering, Colorado State University, Fort Collins, CO 80523; Tel: 970-491-7858, Fax: 970-491-2249, echong@engr.colostate.edu.

[2]This research is supported in part by the National Science Foundation through grants ECS-9501652, NCR-9624525, and ANI-9805441.

II. System Model

We consider a time-slotted system—time is the resource to be shared among all users. Each time-slot is exclusively used by one user within a cell at a given frequency. We use a stochastic model to capture the *time-varying* and *channel-condition-dependent* performance of each user. Specifically, let $\{U_i^k\}$ be the 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, we call this quantity U_i^k the *performance value*. 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 throughput and/or power consumption. This performance value U_i^k for user i may be computed from an estimate of its channel quality. Usually, the better the channel condition of user i, the larger the value of U_i^k. Assuming that the system is stationary, we drop the time index k from our notation. Specifically, we use the notation $\vec{U} = (U_1, \cdots, U_N)$, where U_i is a random variable representing the performance measure of user i at a generic time-slot, and N is the number of users in the cell.

Based on the performance values of the users, the scheduling problem can be stated as: *which user should be assigned which time-slot/channel?* We define a *policy* Q as a mapping from the performance-vector space to the index set $\{1, 2, \cdots, N\}$. Given $\vec{U}$, the policy Q determines the user to be scheduled. The goal of our scheduling policy is to improve wireless resource efficiency by exploiting time-varying channel conditions, while at the same time guaranteeing the minimum performance each user receives.

III. Scheduling with Minimum-Performance Guarantees

A. Scheduling Problem Formulation

In this paper, we study a scheduling problem where the average system performance is maximized subject to meeting each user's minimum-performance requirement. The problem is formulated as:

$$\begin{aligned} \underset{Q}{\text{maximize}} \quad & E\left(U_{Q(\vec{U})}\right) \\ \text{subject to} \quad & E\left(U_i \mathbf{1}_{\{Q(\vec{U})=i\}}\right) \geq C_i, i = 1, \cdots, N, \end{aligned} \quad (1)$$

where $E(U_i \mathbf{1}_{\{Q(\vec{U})=i\}})$ is the average performance of user i, and $C_i \geq 0$ is the minimum performance requirement of user i. We call the vector $\vec{C} = \{C_1, C_2, \cdots, C_N\}$ the *requirement vector.*

The formulation here offers users a direct service guarantee. For example, if the performance measure is defined as the data-rate, then each user is guaranteed a minimum data-rate. While such minimum-performance guarantees is appealing to users, it also raises the feasibility issue—can the system satisfy the performance requirements for all users? In summary, the problem defined in (1) has two aspects:

- Is $\vec{C}$ a feasible requirement vector; i.e., does there exist a policy Q such that $E(U_i \mathbf{1}_{\{Q(\vec{U})=i\}}) \geq C_i$ for all i?
- If $\vec{C}$ is a feasible requirement vector, which policy maximizes the overall performance under the QoS constraint, and how does one implement it?

B. Optimal Scheduling Policy

In this section, we first present the optimal scheduling policy assuming feasibility, and then we discuss the feasibility problem. Suppose $\vec{C} = \{C_1, C_2, \cdots, C_N\}$ is feasible; i.e., there exists some scheduling policy Q such that $E(U_i \mathbf{1}_{\{Q(\vec{U})=i\}}) \geq C_i$ for all i. Then we define a policy Q^* by:

$$Q^*(\vec{U}) = \underset{i}{\text{argmax}}(\alpha_i^* U_i), \quad (2)$$

where the α_i^*'s are chosen so that:

1. $\min_i(\alpha_i^*) = 1$,
2. $E\left(U_i \mathbf{1}_{\{Q^*(\vec{U})=i\}}\right) \geq C_i$ for all i,
3. For any user i, if $E\left(U_i \mathbf{1}_{\{Q^*(\vec{U})=i\}}\right) > C_i$, then $\alpha_i^* = 1$.

Proposition 1: The policy Q^* defined in (2) is a solution to the problem defined in (1); i.e., it maximizes the average system performance under the minimum-performance requirement.

We refer readers to [5] for a proof of the proposition. The policy Q^* maximizes the average system performance even if the users' performance values are arbitrarily correlated, both in time and across users. Note that the parameter $\vec{\alpha}$ "scales" the performance values of users, and the scheduling policy schedules a "relatively-best" user, where user i is relatively-best if $\alpha_i^* U_i = \max_j \alpha_j^* U_j$. If the scale factor for a user is larger than 1, then the user is a "bad" user; i.e., it has to take advantage of some other users (i.e., users with $\alpha_j^* = 1$) to satisfy its performance requirement. Hence, to maximize the overall system performance, a bad user is only granted an average performance value that equals its minimum-performance requirement. When $E(U_i \mathbf{1}_{\{Q(\vec{U})=i\}}) > C_i$ for user i, the user gets more than its minimum requirement, but this user cannot take advantage of other users; i.e., $\alpha_i^* = 1$.

The parameter vector $\vec{\alpha}^*$ in Prop. 1 is determined by the distribution of $\vec{U}$ and the values of the C_i's. In practice, the distribution of $\vec{U}$ is unknown—we use a stochastic approximation algorithm to estimate the value of $\vec{\alpha}^*$. Owing to page limitations, we refer readers to [5] for a detailed discussion on the parameter estimation issue.

Furthermore, our opportunistic scheduling policy dominates non-opportunistic policies in the following sense: Consider a non-opportunistic scheduling policy (one that does not use information on $\vec{U}$) in which user i shares a portion r_i of the resource (time-slots), where $\sum_i r_i = 1$, and user i gets an average performance value $r_i E(U_i)$. Let $C_i = r_i E(U_i)$ for all i. Then $\vec{C}$ is feasible, and the opportunistic scheduling policy always provides "no-loss" performance values for each user relative to that of the non-opportunistic scheduling policy, assuming that the signaling cost is negligible.

In [3], [6], the authors study scheduling algorithms where both delay and channel conditions are taken into account. Roughly speaking, the algorithm is: $\text{argmax}\, b_i W_i t_i$, where W_i is the head-of-the-line packet delay for queue i, t_i is the channel capacity, and b_i is some constant. Furthermore, they have the following result (with more restrictive technical assumptions)—to maximize the system throughput with minimum-throughput requirements, there exists some constant c_i, such that one should choose a user with the maximum value of $c_i t_i$. However, there is no discussion on how to obtain c_i's, how to break ties, or how feasibility plays a role.

The ability to provide a specific performance guarantee is an advantage of our scheme. However, to satisfy such a constraint introduces the question of feasibility. In the following, we discuss the feasibility problem and how to determine the feasibility region of our scheduling policy. The feasibility region of a policy Q is the set of requirement vectors that are feasible under the policy Q.

Proposition 2: The feasible region of our opportunistic scheduling policy is *convex* and contains the feasibility regions of all policies.

A proof is included in [5]. The feasibility region of our scheduling policy is determined by the distribution of $\vec{U}$. In general, there is no closed-form expression for the feasibility region even if the distribution function of $\vec{U}$ is known. The distribution of $\vec{U}$ depends on the user's channel condition, its mobility, and the form of the performance value function. It is practically impossible to know the distribution of $\vec{U}$ *a priori* in the system. Hence, we estimate the feasibility region using sample paths; i.e., the sequence of $\{U_i^k\}$. Convexity is an important feature in determining whether a requirement vector is feasible.

In general, the feasibility region is in an N-dimensional space, where N is the number of users in the system. The vertex on the ith axis is $[0, 0, \cdots, E(U_i), \cdots, 0]$, which corresponds to assigning all the resource to user i. In the extreme case that all the resource is assigned to a single user, there is no performance difference between opportunistic and non-opportunistic scheduling policies. Hence, they have the same vertices on the axes. These N vertices span an $(N-1)$-dimensional hyperplane. Any *non-negative* vector below this hyper-plane is a feasible requirement vector for a non-opportunistic scheduling policy.

The feasibility region of the opportunistic scheduling policy contains the feasibility region of any non-opportunistic scheduling policy, while they share the same vertices on the axes. Next, we determine some other vertices in the feasibility region of the opportunistic scheduling policy. Let

$$Q_{\vec{\alpha}}(\vec{U}) = \underset{i}{\operatorname{argmax}}(\alpha_i U_i), \quad (3)$$

where ties are broken randomly. Given a value of $\vec{\alpha}$, using policy $Q_{\vec{\alpha}}(\vec{U})$ results in an average performance vector, where its ith component is the average performance value of user i (i.e., $E(U_i \mathbf{1}_{Q_{\vec{\alpha}}(\vec{U})=i})$). In other words, by choosing a value for the vector $\vec{\alpha}$, we obtain an average performance-value vector that determines one point on the boundary of the feasibility region. By varying the values of $\vec{\alpha}$, we can draw the boundary of the feasibility region. For example, if we set $\vec{\alpha} = [1, 1, \cdots, 1]$ in (3), then we get the average performance-value vector representing the maximum performance the system can obtain. By using different values of $\vec{\alpha}$ in (3), we get different performance-value vectors, resulting in different points in the N-dimensional space. These points, along with the N vertices in the N axes, span an N-dimensional surface. Because the feasibility region is *convex*, any *non-negative* vector under this surface is feasible. If we choose more values of $\vec{\alpha}$, we get more points on the boundary of the feasibility region, and thus we get a closer approximation to the actual feasibility set.

Figure 1 shows the feasibility region for two users. The performance values of user 1 and user 2 are independent and exponentially distributed with mean values 4 and 5, respectively. The two vertices on the two axes correspond to the two extreme cases that all the resource is assigned to one user. The area between the straight line (a 1-dimensional "plane") and the two axes is the feasibility region of a non-opportunistic scheduling policy. The uppermost dashed curve indicates the boundary of the feasibility region of the opportunistic scheduling policy. The area between the uppermost curve and the two axes is the feasibility region of our opportunistic scheduling policy. The other two curves in between are approximations of the boundary of the feasibility region. The solid line is the case where we have one vertex ($\vec{\alpha} = [1, 1]$)) besides the two vertices on the axes. Note that the vertex corresponding to $\vec{\alpha} = [1, 1]$ is the maximum-system-performance case. By adding two more points ($\vec{\alpha} = [1, 3]$ and $\vec{\alpha} = [3, 1]$), we get a better approximation (dotted curve with stars), which gives a fairly close approximation of the feasibility region. Hence, it is fair to say that we can obtain reasonable estimates the feasibility region via measurement data.

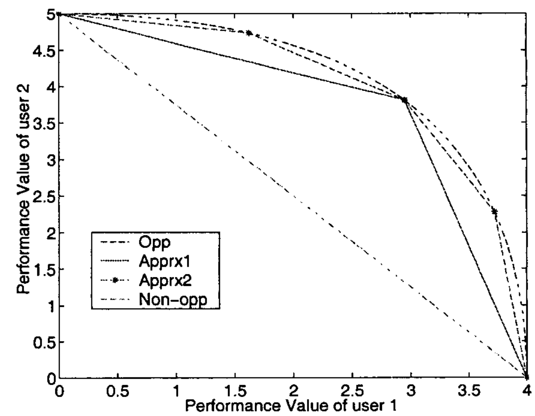

Fig. 1. The feasible region of two users.

Next, we discuss how to determine the feasibility region when the set of (active) users changes in the cell. Suppose there are currently N users in the system, and we have some information about these N users. We can estimate the feasibility region for the N users using different values of $\vec{\alpha}$. Note that we can draw these vertices simultaneously via measurements of $\vec{U}$. We consider two situations: First, if a user leaves the system, we simply collapse the feasibility set from N dimensions to $N-1$ dimensions by removing the axis of the leaving user. Second, suppose a new user joins the system, and we do not have any information on the user except its average performance value $E(U_{N+1})$. We can connect all the points on the surface of the feasibility set for the original N users with the new vertex on the $(N+1)$th axis, and construct a surface of $N+1$ dimensions. Due to the convexity, any non-negative $N+1$ dimensional vector under the new surface is feasible.

Compared with non-opportunistic schedulers, opportunistic schedulers *enlarge* the feasible region/capacity. Hence, we can achieve the following goals:

1. Accommodate more users under the same admission control policy. For example, consider a simple admission control algorithm—users are admitted in the system as long as the requirement vector is feasible; i.e., the admission region is the same as the feasibility region. As shown in Figure 1, the uppermost curve indicates the boundary of the feasibility region of the opportunistic scheduling policy, whereas the straight (lowest) line is that of a non-opportunistic scheduling policy. The area between these two curves is the area where the opportunistic scheduling policy can accommodate two users while the non-opportunistic scheduling policy can only accommodate one user.
2. Improve users' service quality, in terms of higher performance and/or lower degradation probability, when the same users are admitted in a non-opportunistic scheduling scheme. Degradation means that the system cannot satisfy the performance "guarantee" given to a user at admission due to system

load changes (say, handoff), variation in channel conditions, and mobility. If we choose feasible C_i's and let $C_i \geq r_i E(U_i)$, then our opportunistic scheduling policy results in a "no-loss" situation for each user over non-opportunistic scheduling policies, which guarantees better service quality for users.

IV. Simulation Results

In this section, we present simulation results 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 and the minimum-performance requirement.

For the purpose of comparison, we also simulate two other scheduling policies. The first is round-robin, a non-opportunistic scheduling policy, which schedules (active) users following a predetermined order. This scheduling scheme serves as a benchmark of the system performance. As a comparison, we measure how much gain the system can obtain using opportunistic scheduling policies. The second is a greedy scheduling policy, which always selects the user with the maximum performance value to transmit. This policy achieves the maximum performance the system can obtain without any constraints, thereby providing an upper bound on the achievable performance. In the following, we first describe the simulation model of a cellular cell, and then we show the simulation result for the scheduling policy using the cellular model.

A. Cellular Model

In our simulation, we consider a multi-cell 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 omni-directional antennas are used by mobiles and base stations. We focus on the performance of the downlink of the center cell because downlink communication is more important for data services. The frequency reuse factor is 3, and the co-channel interference from the six first-ring neighboring cells are taken into account. We assume that each cell has a fixed number of frequency bands. We focus on one frequency band that is shared by 10 users in the central cell. The scheduling policy decides which user should transmit in this frequency band at each time-slot. The users have exponentially distributed "on" and "off" periods.

Users move with random speed and direction in the cell. 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 shadowing model in [7]. To be conservative, we ignore the effects of fast multi-path fading in the simulation. If fast fading could be accurately tracked, 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].

Figure 2 shows the forms of the performance values used by different users (there are 10 users in the system). The performance values of users 1, 5, and 8 are step-functions of their SINR. The performance values of users 2, 6, and 9 are linear functions of their SINR (in dB). Users 3, 4, 7, and 10 have performance values that are S-shape functions of their SINR. Furthermore, the 10 users are divided into three "distance" groups; i.e., when a user becomes active, its distance from the base station is fixed, depending on which group it belongs to. Users 1–4 belong to the "far" group; i.e., when the user becomes active, its distance from the base station is $0.9R$, where R is the radius of the cell. Users 5–7 belong to the "middle" group; their starting distance from the base station is $0.5R$. Users 8–10 belong to the "near" group with a starting distance $0.2R$. Note that when the user is active, it moves around in the cell freely and randomly. However, a user in "near" group has much larger chance to be close to the base station than a user in "far" group. Hence, we can study how the distance from the base station effects users' performance in the different scheduling schemes.

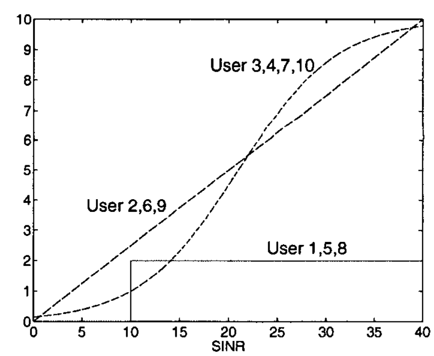

Fig. 2. Users' performance as a function of SINR.

The system performs the same procedure at every time-slot. Basically, the users measure their channel conditions, calculate their performance values, and then send the information to the base station. The base station decides the user to be scheduled using the opportunistic scheduling policy, based on the information of the users' performance values. Then the base station updates the parameter used in the scheduling policy. A detailed simulation procedure is omitted due to the page limit, and we refer interested readers to [5].

B. Numerical Results

In the following, we show simulation results for the opportunistic scheduling scheme with minimum-performance guarantees. First, we run the simulation for 1,000,000 time-slots using the round-robin scheduling policy, where the resource is equally distributed among all users, and active users are scheduled in a predetermined order. Then we get an average performance value for each user. We use it as the minimum-performance requirement for each user. Then we run a simulation using the opportunistic scheduling policy, the round-robin policy, and the greedy scheduling policy.

Figure 3 shows the average performance values of users resulting from the different scheduling policies. The first bar indicates the average performance values using the round-robin scheduling policy, the second bar is the minimum-performance requirement of a user, the third bar indicates the result from our scheduling policy, and the rightmost bar is that of the greedy algorithm. Note that our scheduling policy outperforms the round-robin policy uniformly, which illustrates the "no-loss" situation discussed earlier in Section III-B. Compared with round-robin, our scheduling policy improves the overall system performance by 51% while the greedy scheduling has an improvement of 109%. It is not surprising that the greedy algorithm results in the highest overall performance value at the cost of the extreme unfairness among users (e.g., under this strategy users 1 and 8 never get a chance to transmit).

Figure 4 indicates the amount of resource consumed by each user in different scheduling policies. The first bar represents that of round-robin, the second bar indicates our scheduling policy, and the third bar is the greedy scheduling.

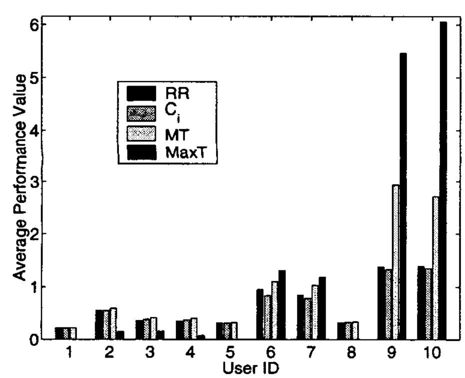

Fig. 3. Average performance value in the minimum-performance guarantee scheduling simulation.

In summary, the simulation shows that using our scheduling policy, the system can achieve significant performance gains while satisfying the minimum-performance requirement. Because we set the requirement to be the performance value obtained from round-robin, all users perform better using our policy than in round-robin.

V. Conclusions

Opportunistic scheduling is one way to improve spectrum efficiency by exploiting time-varying channel conditions. In this paper, we study an opportunistic scheduling problem—to maximize the average system performance by exploiting variations of the channel conditions while satisfying the minimum-performance requirements of users. We present an optimal solution and discuss the feasibility problem.

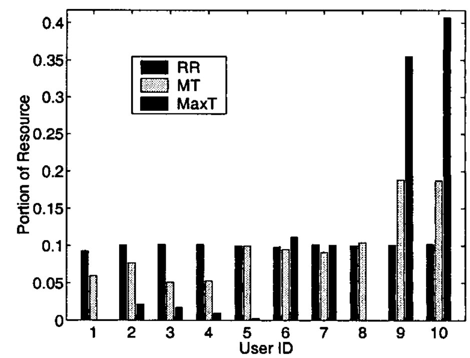

Fig. 4. Portion of resource shared by users in the minimum-performance guarantee scheduling simulation.

The assumptions in our study are not significantly restrictive: we assume that the performance values of users (i.e., $\{\vec{U}^k\}$) are stationary but can have arbitrary correlations across time or across users. Furthermore, we also provide simulation results for a realistic cellular system, where users are moving randomly, and have on-off activity periods so that the stationary assumption is in general not guaranteed. Simulation results show that our scheme performs well, and can track the changes of the system.

Our scheduling scheme is easy to implement. At each time-slot, the base station schedules a relatively-best user to transmit. The base station also updates the parameters used in the scheduling policy. The calculation burden per time-slot is $O(N)$, where N is the number of users in a cell, which is usually on the order of tens.

References

[1] 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 Communications Magazine*, pp. 70–77, July 2000.

[2] 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.

[3] M. Andrews, K. Kumaran, K. Ramanan, A. Stolyar, P. Whiting, and R. Vijayakumar, "Providing quality of service over a shared wireless link," *IEEE Communications Magazine*, pp. 150–153, February 2001.

[4] X. Liu, E. K. P. Chong, and N. B. Shroff, "Transmission scheduling for efficient wireless network utilization," *IEEE Journal on Selected Areas in Communications*, in press, 2001, http://shay.ecn.purdue.edu/~xinliu/papers.html.

[5] X. Liu, E. K. P. Chong, and N. B. Shroff, "Opportunistic scheduling for efficient wireless network utilization with QoS constraints," 2000, http://shay.ecn.purdue.edu/~xinliu/papers.html.

[6] M. Andrews, S. Borst, F. Dominique, P. Jelenkovic, K. Kumaran, K. Ramakrishnan, and P. Whiting, "Dynamic bandwidth allocation algorithms for high-speed data wireless networks," *Bell Labs Technical Report*, 2000.

[7] G. Stuber, *Principles of Mobile Communication*, Kluwer Academic Publishers, 1996.