The IMA Volumes in Mathematics and its Applications

Volume 143

Series Editors
Douglas N. Arnold Arnd Scheel

Institute for Mathematics and its Applications (IMA)

The **Institute for Mathematics and its Applications** was established by a grant from the National Science Foundation to the University of Minnesota in 1982. The primary mission of the IMA is to foster research of a truly interdisciplinary nature, establishing links between mathematics of the highest caliber and important scientific and technological problems from other disciplines and industries. To this end, the IMA organizes a wide variety of programs, ranging from short intense workshops in areas of exceptional interest and opportunity to extensive thematic programs lasting a year. IMA Volumes are used to communicate results of these programs that we believe are of particular value to the broader scientific community.

The full list of IMA books can be found at the Web site of the Institute for Mathematics and its Applications:

http://www.ima.umn.edu/springer/volumes.html

Douglas N. Arnold, Director of the IMA

* * * * * * * * * *

IMA ANNUAL PROGRAMS

1982–1983	Statistical and Continuum Approaches to Phase Transition
1983–1984	Mathematical Models for the Economics of Decentralized Resource Allocation
1984–1985	Continuum Physics and Partial Differential Equations
1985–1986	Stochastic Differential Equations and Their Applications
1986–1987	Scientific Computation
1987–1988	Applied Combinatorics
1988–1989	Nonlinear Waves
1989–1990	Dynamical Systems and Their Applications
1990–1991	Phase Transitions and Free Boundaries
1991–1992	Applied Linear Algebra
1992–1993	Control Theory and its Applications
1993–1994	Emerging Applications of Probability
1994–1995	Waves and Scattering
1995–1996	Mathematical Methods in Material Science
1996–1997	Mathematics of High Performance Computing
1997–1998	Emerging Applications of Dynamical Systems
1998–1999	Mathematics in Biology

Continued at the back

Prathima Agrawal Daniel Matthew Andrews
Philip J. Fleming George Yin
Lisa Zhang
Editors

Wireless Communications

Springer

Prathima Agrawal
Department of Electrical and
Computer Engineering
Auburn University
Auburn, AL 36849-5201
USA
www.eng.auburn.edu/~pagrawal

Daniel Matthew Andrews
Bell Laboratories
Lucent Technologies
Murray Hill, NJ 07974-0636
USA
cm.bell-labs.com/cm/ms/who/andrews/

Philip J. Fleming
Network Advanced Technology
Motorola, Inc.
Arlington Heights, IL 60004
USA

George Yin
Department of Mathematics
Wayne State University
Detroit, MI 48202
USA
www.math.wayne.edu/~gyin/

Lisa Zhang
Computing Sciences
Research Center
Bell Laboratories
Lucent Technologies
Murray Hill, NJ 07974
USA
cm.bell-labs.com/who/ylz/

Series Editors
Douglas N. Arnold
Arnd Scheel
Institute for Mathematics and its
Applications
University of Minnesota
Minneapolis, MN 55455
USA

Mathematics Subject Classification (2000): 90B18, 94-06, 94A05, 60G35

Library of Congress Control Number: 2006933293

ISBN-10: 0-387-37269-5
ISBN-13: 978-0387-37269-3

Camera-ready copy provided by the IMA.

9 8 7 6 5 4 3 2 1

springer.com

FOREWORD

This IMA Volume in Mathematics and its Applications

Wireless Communications

contains papers based on invited lectures at the very successful IMA Summer Program on Wireless Communications, held on June 22 – July 1, 2005. We would like to thank Prathima Agrawal (Auburn University), Daniel Matthew Andrews (Lucent Technologies), Philip J. Fleming (Motorola, Inc.), George Yin (Wayne State University), and Lisa Zhang (Lucent Technologies) for their superb role as workshop organizers and editors of the proceedings.

We take this opportunity to thank the National Science Foundation for its support of the IMA.

Series Editors

Douglas N. Arnold, Director of the IMA

Arnd Scheel, Deputy Director of the IMA

PREFACE

This volume presents papers, based on invited talks given at the 2005 IMA Summer Workshop on Wireless Communications, held at the Institute for Mathematics and Its Applications, University of Minnesota, June 22 – July 1, 2005.

The conference provided a well blended program to facilitate the communications between academia and the industry, and to bridge the mathematical sciences, engineering, information theory, and communication communities. The emphases were on design and analysis of computationally efficient algorithms to better understand the behavior and to control the wireless telecommunication networks. As an achieve, this volume presents some of the highlights of the conference, and collects papers covering a broad spectrum of topics. All papers have been reviewed.

Without the help, assistance, support, and encouragement of many people, this workshop could not come into being. We thank the invited speakers, the poster presenters, and all attendees for making the conference a successful event. Our thanks go to Douglas N. Arnold and Fadil Santosa for helping us shaping the conference and proving us with valuable comments and suggestions. We are grateful to Debra Lewis and the IMA staff for their tireless help in the preparation stage and during the conference. The assistance from Arnd Scheel in preparing the proceedings is gratefully acknowledged. We also thank Patricia V. Brick and Dzung N. Nguyen for their help and assistance for putting the final product together in a beautiful piece.

Prathima Agrawal
Department of Electrical and Computer Engineering
Auburn University

Daniel Matthew Andrews
Bell Laboratories
Lucent Technologies

Philip J. Fleming
Network Advanced Technology
Motorola, Inc.

George Yin
Department of Mathematics
Wayne State University

Lisa Zhang
Bell Laboratories
Lucent Technologies

CONTENTS

A SURVEY OF SCHEDULING THEORY IN WIRELESS DATA NETWORKS

MATTHEW ANDREWS*

Abstract. We survey some results for scheduling data in wireless data systems such as 1xEV-DO. An important feature of such systems is that the channel rates between the basestation and the mobile users are both user-dependent and time-varying. The wireless data scheduling problem has recently received a great deal of attention in the literature. However, comparisons of results are sometimes difficult due to the fact that many different models have been studied. In this survey we describe some of the models that have been proposed and analyze the performance of different scheduling algorithms within these models.

1. Introduction. The advent of third-generation wireless systems such as CDMA2000 1xEV-DO [15, 18] means that mobile Internet users can now obtain high data rates in cellular systems. However, in order to effectively utilize the wireless capacity, we require efficient methods for deciding how the wireless resources should be assigned. In particular, we require *scheduling* algorithms that determine which user should be served in each time step.

In this paper we survey some of the basic results about scheduling in wireless data systems. The most important feature of such systems is that due to channel fading and user mobility, the rates at which users can receive data are both *user-dependent* and *time-varying*. In particular, when a user is close to the transmitter it can typically receive data at a higher rate than when it is farther away from the the transmitter. In recent years, there has been a great deal of work on developing effective scheduling algorithms for wireless data systems. Unfortunately, many of the results in the literature consider different models which makes comparing results difficult. In this survey, we define a number of models that have been studied and describe what scheduling results are known in each of them. We begin by describing in detail the basic scheduling problem together with the different models that have been considered.

The model. We consider a set of n mobile data users in a wireless cell served by a single basestation. (See Figure 1). We focus on the downlink (basestation to mobile) direction since for many applications such as web browsing the majority of data flows in that direction. The basestation maintains a separate queue of data at the basestation for each mobile user. Time is slotted and in each time slot the basestation can transmit data to exactly one user. In order to make this decision the basestation knows at all time steps a vector $(r_0(t), \ldots, r_{n-1}(t))$, where $r_i(t)$ is the amount of data that can be transmitted to user i at time step t. In the EV-DO system this value is known as the Data Rate Control (DRC) value. As already

*Bell Laboratories, Murray Hill, NJ 07974 (andrews@research.bell-labs.com).

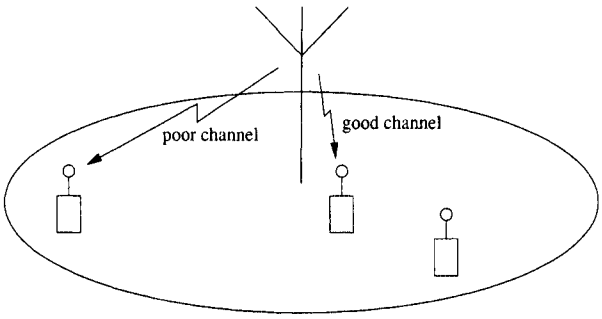

FIG. 1. *A wireless system.*

mentioned, channel fading and user mobility mean that DRC values are *user-dependent* and *time-varying.*

The scheduler at the basestation knows the value of $r_i(t)$ because at each time step mobile user i measures the strength of a pilot signal transmitted by the basestation. From the strength of that signal user i can calculate the quality of the channel between the basestation and itself and determine the rate at which the basestation should transmit in order to achieve a low error. The user then sends this rate to the basestation in a control message.

The time-varying nature of the channel rates makes the scheduling problem much more complex than in the wireline setting since the "correct" decision about which user to serve will change from time slot to time slot. In general we want to "ride the peaks" of the channel processes and try to pick a user whose current channel condition is better than average. On the other hand, we want to schedule fairly and not starve any users whose channel conditions are poor.

Formally, the scheduling problem is as follows. In each time step the scheduler receives the channel rate vector $(r_0(t), \ldots, r_{n-1}(t))$. It then makes a decision about which user to serve. If user i is chosen then $r_i(t)$ bits are served from the queue of user i (or all the data is served if the queue size is less than $r_i(t)$). As already mentioned, one of the aims of this survey is to highlight the differences between some of the models that are considered in the literature. The models differ in the assumptions that are made about the arrival model and in the assumptions that are made about the channel process between the basestation and the mobile users.

With regards to the traffic model, the two options usually considered are an infinitely-backlogged model and a model where the queues for each user are fed by an external arrival process. More formally, these models are defined as follows:

- In the infinitely backlogged model each user always has data to transmit. Since there is no arrival process as such, metrics such as queue size and delay do not make much sense. We wish to

optimize some function of the throughputs achieved by the users. For example, if R_i is some measure of the long-term throughput achieved by user i, a popular goal is to optimize the *Proportional Fair* metric $\sum_i \log R_i$.

- In the model with an external arrival process, at each time step the scheduler receives a vector $(a_0(t), \ldots, a_{n-1}(t))$, where $a_i(t)$ represents the amount of data that arrives for user i in time slot t. This vector is in addition to the channel rate vector described above. In this case, metrics such as queue size and delay become relevant in addition to throughput. One fundamental goal of any scheduling algorithm is *stability*. In particular, an essential attribute that we would like a scheduler to possess is *stability*. We say that a scheduler is stable if it keeps the queue sizes bounded whenever this is feasible.

With regards to the channel process the two models that are studied are a model in which the channel rates are generated according to a stationary stochastic process and a worst-case model in which the channel rates are generated by an adversary:

- For the model in which the channel rates are generated according to a stationary stochastic process we assume that there is a finite set of (aggregate) channel states denoted by $M = \{1, \ldots, M\}$. Associated with each state $m \in M$ is a fixed vector of data rates $(\mu_1^m, \ldots, \mu_N^m)$. The channel process is defined by an ergodic Markov Chain $m(t)$ with state space M. In particular, whenever $m(t) = m$ the channel rate vector is given by $r_i(t) = \mu_i^m$.
  In this model we typically aim to derive the "optimal" scheduling rule with respect to some metric. It is often convenient to compare a candidate scheduling algorithm against an ideal *Static Service Split* (SSS) rule in which we have a set of ϕ_{mi} such that $\sum_i \phi_{mi} = 1$. Whenever the state of the Markov Chain is m the SSS rule serves user i with probability ϕ_{mi}. We note however that it is typically not feasible to implement the optimal SSS rule since the scheduler is not aware of the structure of the underlying Markov Chain.
- For the adversarial model we do not assume any type of stationarity. Instead at each time step t the channel rate vector $(r_0(t), \ldots, r_{n-1}(t))$ can be an arbitrary vector that is defined by an *adversary*. We can think of the adversary as trying to create as much trouble for the scheduling algorithm as possible. As in the stationary model we typically wish to compare a candidate online scheduling algorithm against an "ideal" algorithm. The SSS rules do not make much sense here because the optimal scheduling decision for a particular rate vector may change over time. Instead, we assume that at each time step, the adversary has its own schedule that will produce good performance in conjunction with

the channel rate vectors that it generates. Our aim is to match the adversary's schedule as closely as possible.

By combining the two possible traffic arrival models with the two channel models we obtain four possible models, each of which has been studied in the literature. In the next four sections of the paper we consider each model separately and present some of the results that are known. We then briefly discuss the case of wireless mesh networks in which there are multiple transmitters and data may need to pass through more than one node.

2. Infinitely backlogged queues - Stationary channel process. In the first model that we consider we assume that all users always have data to send and that the channel conditions are generated by a stationary stochastic process. This is the model that generated one of the most widely used wireless scheduling algorithms, namely the Proportional Fair scheduling algorithm of Tse [28]. In each time step Proportional Fair serves user,

$$j = \arg\max_i \frac{r_i(t)}{R_i(t)},$$

where $R_i(t)$ is the value at time t of an exponentially filtered average service rate that is updated according to,

$$R_i(t+1) = \begin{cases} (1-\tau)R_i(t) + \tau r_i(t) & \text{if } i = j \\ (1-\tau)R_i(t) & \text{if } i \neq j \end{cases}$$

for some time constant τ. (In practice τ is typically on the order of 1000 slots.) Note that the Proportional Fair algorithm gives priority to users with a high instantaneous channel value ($r_i(t)$) and a low current average service rate ($R_i(t)$).

The Proportional Fair algorithm has an extremely elegant theoretical property. It maximizes, over all feasible scheduling rules, the function $\sum_i \log R_i$, where R_i is the long-term service rate of user i. This objective is sometimes known as the Proportional Fair metric for the following reason. If $(R_0^*, .., R_{n-1}^*)$ is the vector of feasible rates that maximizes $\sum_i \log R_i$, then for any other vector of rates $(R_0, \ldots, R_{n-1})$, we have that $\sum(R_i - R_i^*)/R_i^* < 0$. In other words, if we move from R_i^* to another feasible rate allocation and we scale the improvement for user i in proportion to the current allocation, the aggregate improvement must be negative. Another way to look at the metric is that multiplying one user's rate by a factor c has the same effect on the objective as multiplying another user's rate by the same factor c. Lastly, observe that by using $\sum_i \log R_i$ as a metric we do not starve any user completely since $\log 0 = -\infty$.

At a high-level the reason why Proportional Fair is optimal with respect to the metric $\sum_i \log R_i$ is as follows. If we let $S(t) = \sum_i \log R_i(t)$ then $(\nabla S)(t) = (\frac{1}{R_0(t)}, \ldots, \frac{1}{R_{n-1}(t)})$. We then have,

$$\begin{aligned}
&\sum_i \log R_i(t+1) - \sum_i \log R_i(t) \\
&\approx (\nabla S)(t) \cdot (R_0(t+1) - R_0(t), \ldots, R_{n-1}(t+1) - R_{n-1}(t)) \\
&= \left(\frac{1}{R_0(t)}, \ldots, \frac{1}{R_{n-1}(t)}\right) \cdot (R_0(t+1) - R_0(t), \ldots, R_{n-1}(t+1) - R_{n-1}(t)) \\
&= \left(\frac{1}{R_0(t)}, \ldots, \frac{1}{R_{n-1}(t)}\right) \cdot (-\tau R_0(t), \ldots, \tau r_j(t) - \tau R_j(t), \ldots, -\tau R_{n-1}(t)) \\
&= \frac{\tau r_j(t)}{R_j(t)} - \tau n,
\end{aligned}$$

whenever user j is served. Since τ and n are fixed this implies that in order to maximize the change in $\sum_i \log R_i(t)$ we should serve the user that maximizes $r_i(t)/R_i(t)$. This is exactly what Proportional Fair does.

A formal proof of the optimality of Proportional Fair has been obtained in multiple contexts (e.g. [20, 1, 25]. In this survey we state the asymptotic result of Stolyar [25]. Let R_i^* be the steady-state rate allocation of the optimal SSS rule with respect to the metric $\sum_i \log R_i$. Consider a sequence of processes indexed by $\tau_0, \tau_1, \tau_2, \ldots$, where $\tau_k \downarrow 0$. For each τ_k we have a fixed initial state $R_i^{\tau_k}(0)$ for the average service rates and a fixed initial state $m^{\tau_k}(0)$ for the Markov Chain that governs the channel process. Suppose that the channel rates evolve according to the Markov Chain and the average service rates evolve according to the Proportional Fair algorithm.

Let $D_i^{\tau_k}(t)$ be the amount of service received by user i in the time slot t under the process indexed by τ_k. Let $R_i^{\tau_k}(\ell_1, \ell_2)$ be the average service rate received between time slot ℓ_1 and time slots ℓ_2, namely,

$$R_i^{\tau_k}(\ell_1, \ell_2) = \frac{1}{\ell_2 - \ell_1 + 1} \sum_{t=\ell_1}^{\ell_2} D_i^{\tau_k}(t).$$

Then, the main theorem of Stolyar [25] implies,

THEOREM 2.1. *Let A be a bounded subset of R_+^n. Then, for any $\varepsilon > 0$, there exist parameters T_1 and T_2 (both depending on A and ε) such that,*

$$\lim_{\tau_k \downarrow 0} \sup_{R_i^{\tau_k}(0) \in A, \ell_1 > T_1/\tau_k, \ell_2 > T_2/\tau_k} \sqrt{(E[R_i^{\tau_k}(\ell_1, \ell_2)] - R_i^*)^2} \leq \varepsilon,$$

where $E[\cdot]$ denotes expectation. In other words, as $\tau \downarrow 0$ the long-term average service rate approaches the average service rate of the optimal SSS rule.

Another interesting property of Proportional Fair is that if the channel processes take the form $r_i(t) = a_i \cdot b_i(t)$, where a_i is a constant and the $b_i(t)$ processes are i.i.d. then in the long run the fraction of slots allocated to each user is $1/n$. That is, if the channel rate fluctuations around the mean are the same for all users then the scheduler serves each user for an equal amount of time. This property was derived by Holtzman in [17].

Proportional Fair with Minimum/Maximum Rate Constraints. Note that although the Proportional Fair algorithm maximizes the metric $\sum_i \log R_i$, it does not provide any absolute guarantees on the service rate provided to any individual user. For some applications, e.g. streaming video, we may need to provide a minimum bandwidth to the users in order for the application to be useful. In some cases we may also want to limit the amount of service that a user receives, e.g. if we want to encourage a user to upgrade to a more expensive service. Suppose therefore that for each user i we have a minimum rate $R_i^{\min}$ and a maximum rate $R_i^{\max}$ and we want the average service rate R_i to satisfy, $R_i^{\min} \leq R_i \leq R_i^{\max}$. The optimization problem then becomes,

$$\begin{aligned} &\max \sum_i \log R_i \\ &\text{subject to} \\ &\qquad R_i^{\min} \leq R_i \leq R_i^{\max} \end{aligned} \tag{2.1}$$

An algorithm for this problem called Proportional Fair with Minimum/Maximum Rate Constraints (PFMR) was presented in [9]. The algorithm operates by maintaining a token counter $T_i(t)$ for each user i. The role of this token counter is to enforce the rate constraints. It is updated according to

$$T_i(t+1) = \begin{cases} T_i(t) + R_i^{token} - r_i(t) & \text{if user } i \text{ is served} \\ T_i(t) + R_i^{token} & \text{otherwise} \end{cases}$$

where $R_i^{token} = R_i^{\min}$ if $T_i(t) \geq 0$ and $R_i^{token} = R_i^{\max}$ if $T_i(t) < 0$. At time step t the PFMR algorithm serves the user,

$$j = \arg\max_i \frac{r_i(t)}{R_i(t)} e^{a_i T_i(t)},$$

where a_i is a parameter that determines the timescale over which the rate constraints are satisfied. The basic idea of the token counter is that if the average service rate to user i is less than $R_i^{\min}$ then $T_i(t)$ becomes positive and so we are more likely to serve user i. If the average service rate to user i is more than $R_i^{\max}$ then $T_i(t)$ becomes negative and so we are less likely to serve user i. Recall that τ is the time constraint of the exponential filter used to define $R_i(t)$. The paper [9] shows that if $\tau \downarrow 0$ and $a_i \propto \tau$ for all i then as long as the algorithm converges it converges to the optimal solution of the problem (2.1).

We remark that in [21], Liu, Chong and Shroff considered a similar problem to (2.1) and presented a different algorithm that is based on the theory of stochastic approximation.

3. External arrival process - Stationary channel process. The results described in the previous section show that by using the Proportional Fair algorithm we can achieve fair rate allocations in the case that

each user always has data to serve. However, in some situations different users have different amounts of data to serve and our goal is to serve all the data. Suppose that for each user traffic arrives according to some stationary random process. Let $a_i(t)$ be the amount of data arriving for user i at time t. We sometimes refer to the process defined by $(r_i(t), a_i(t))$ as the *input process* of the system. Let $q_i(t)$ be the amount of user i data waiting for service at time t. The queueing process is updated as follows. If user i is served at time t then:

$$q_i(t+1) = \Big[q_i(t) + a_i(t) - r_i(t)\Big]^+,$$

else

$$q_i(t+1) = q_i(t) + a_i(t).$$

Let λ_i be the mean arrival rate for user i. We say that the input process is *schedulable* if there is an SSS rule under which the average long-term service rate R_i is greater than $(1-\varepsilon)\lambda_i$ for some $\varepsilon > 0$. We would like the scheduling algorithm to be *stable*. That is, we would like the algorithm to ensure that the queue process has a stationary distribution whenever the input process is schedulable. Note that if the queue process has a stationary distribution then the aggregate queue size cannot drift to infinity.

3.1. Proportional fair is unstable. An extremely natural question is: "Is the Proportional Fair scheduling algorithm stable?" This question was studied in [3]. However, before we present the answer we must examine exactly how the Proportional Fair algorithm is defined in the case that some of the queue sizes are extremely small. We may not want to serve a user that only has a small amount of data to serve. In [3], three options were considered for whether or not a user is eligible for service.

- Option A1. All users are eligible for service at every time slot.
- Option A2. User i is only eligible for service at time slot t if $q_i(t) > 0$.
- Option A3. User i is only eligible for service at time slot t if $q_i(t) \geq r_i(t)$; i.e. there is enough data to fully utilize the time slot.

Among all eligible users, the one with the highest value of $r_i(t)/R_i(t)$ is selected for service. However, this still does not entirely define the algorithm since there remains the question of how we update the average service rate $R_i(t)$ when the amount of data in the queue of the served user is less than the instantaneous service rate. In [3] the following options were considered,

- Option B1. When user i is served then $R_i(t)$ is updated by $R_i(t+1) = (1-\tau)R_i(t) + \tau r_i(t)$.
- Option B2. When user i is served then $R_i(t)$ is updated by $R_i(t+1) = (1-\tau)R_i(t) + \tau \min\{r_i(t), q_i(t)\}$.

(We remark that as far as we are aware most practical implementations of Proportional Fair use options A2 and B1.)

By considering all possible combinations of the "A" and "B" options we obtain six different algorithms. The main result of [3] is that none of these six algorithms are stable.[1]

The instability example is extremely simple and consists of two users. The arrival process is constant, $a_1(t) = 49$ and $a_2(t) = 94$ for all t. The channel process for user 2 is constant, $r_2(t) = 100$ for all t. The channel process for user 1 is periodic with period 10, namely,

$$r_1(t) = \begin{cases} 1000 & \text{if } t \mod 10 = 0 \\ 100 & \text{otherwise.} \end{cases}$$

This example is schedulable since we could schedule user 1 whenever $t \mod 20 = 0$ and we could schedule user 2 in all other slots. In other words, half of the slots where $r_1(t) = 1000$ are assigned to user 1, all other slots are assigned to user 2. This would result in an average service rate to user 1 of 50 and an average service rate to user 2 of 95. These service rates are more than the respective arrival rates.

However, it is shown in [3] that Proportional Fair is not able to make the correct slot assignments. In particular, for each of the six versions of Proportional Fair, it is shown that user 2 receives only 9 out of 10 slots and hence its average service rate is only 90. Since the arrival rate for user 2 is 94 this means that the queue for user 2 grows without bound.

3.2. Max-weight is stable. Since the Proportional Fair algorithm is not stable, the next question to ask is whether or not there *exists* a stable algorithm. The basic problem with Proportional Fair is that it does not take into account the queue lengths and so it does not know how to react when one queue starts to get too large. A simple algorithm that does not suffer from this problem is the *Max-Weight* algorithm that always serves the user with the maximum value of $q_i(t)r_i(t)$. Note that this algorithm favors users with large instantaneous channel rates and users with large queues. Various analyses have shown that this algorithm is stable, for example [26, 27, 8, 7, 22, 19].

At a high level, the reason why the Max-Weight algorithm is stable is that $||q(t)||$ has a negative drift, where $||q(t)|| = \sqrt{\sum_i (q_i(t))^2}$. To see why this is true, let $x_i(t)$ be the amount of service that user i receives at time t under Max-Weight and let $y_i(t)$ be the amount of service that user i receives at time t under the optimal SSS rule. By the definition of the Max-Weight we have that $\sum_i q_i(t)x_i(t) \geq \sum_i q_i(t)y_i(t)$ and by the definition of

[1] We remark that the instability example of [3] does not quite fit into the model that we have defined in this paper since the channel process is periodic rather than ergodic. (However, we conjecture that the result could also be extended to an example with an ergodic channel process.)

schedulability we have that for large w, $E[\sum_{t \leq t' \leq t+w}(y_i(t') - a_i(t'))] > \varepsilon w \lambda_i$ for all i and for all t. Therefore,

$$\begin{aligned} ||q(t+1)||^2 &= \sum_i (q_i(t) + a_i(t) - x_i(t))^2 \\ &= ||q(t)||^2 + \sum_i q_i(t)(a_i(t) - x_i(t)) + \sum_i (a_i(t) - x_i(t))^2 \\ &\leq ||q(t)||^2 + \sum_i q_i(t)(a_i(t) - y_i(t)) + \sum_i (a_i(t) - x_i(t))^2. \end{aligned}$$

Since for all t and for large w, $E[\sum_{t \leq t' \leq t+w}(y_i(t') - a_i(t'))] > \varepsilon w \lambda_i$, we can use the above inequality to show that $||q(t)||$ has a negative drift over long timescales whenever some queue becomes large. We omit the details from this survey.

4. External arrival process - Adversarial channel process. Proportional Fair and Max-Weight are simple, appealing algorithms with well-defined provable properties. However, all of the results mentioned in the previous two sections make the assumption that the channel process can be modeled by a stationary stochastic process. There may be some situations however where this stationarity assumption does not hold. In particular, consider a vehicle driving away from a basestation. In this case the channel has a negative drift. Hence it makes sense to also consider an adversarial process that allows us to do worst case analysis.

In this section we consider the scenario in which the data is generated by an external arrival process. In this case we assume that the adversary generates the arrival process as well as the channel process. As is usual in adversarial analyses, we must make some restrictions on the adversary otherwise it could overload the system and prevent any algorithm from having reasonable performance.

We therefore define the adversarial model as follows. At each time step t the adversary generates the channel rate for user i, $r_i(t)$, and the amount of arriving data for user i, $a_i(t)$. In order to define whether this situation is schedulable, we assume that the adversary has its own "hidden" schedule. If user i is served by this hidden schedule at time t then we write $z_i(t) = 1$, else $z_i(t) = 0$. We say that the input process is schedulable with parameters (w, ε) if for any sequence of w time steps, the amount of data that arrives for user i in those time steps is less by a factor $(1 - \varepsilon)$ than the amount of service that the hidden schedule gives to user i, i.e. for any t_0, $\sum_{t=t_0}^{t_0+w} a_i(t) \leq (1 - \varepsilon) \sum_{t=t_0}^{t_0+w} r_i(t) z_i(t)$.

4.1. Tracking algorithm is stable. Ideally we would like a schedule such that if the input process is schedulable then the queue sizes are bounded, i.e. there a exists a B such that $q_i(t) \leq B$ for all i, t. This question was addressed in the paper [11]. The first part of this paper looks at impossibility results. In particular, two results were proved. In order

to understand the meaning of these results we let $\mathcal{R}$ be the set of channel rates used by the adversary. We also let $\mathcal{R}^{\inf} = \inf\{r \in \mathcal{R} : r > 0\}$ and $\mathcal{R}^{\sup} = \sup\{r \in \mathcal{R}\}$. Paper [11] shows:

- If $\varepsilon > 0$ then for any online scheduling algorithm A, the adversary can create a schedulable input process such that some queue is unbounded under algorithm A. In this example the rate set is infinite and satisfies $\mathcal{R}^{\inf} = 0$, i.e. the nonzero rates used by the adversary can be arbitrarily small.
- If $\varepsilon = 0$ then for any online scheduling algorithm A, the adversary can create another schedulable input process such that some queue is unbounded under algorithm A. In this example the rate set is infinite and satisfies $\mathcal{R}^{\inf} > 0$, i.e. the nonzero rates used by the adversary are bounded away from zero.

The intuition behind these results is that at each time step the adversary can determine which user will be served by algorithm A at the next time step. It then injects data in such a way that the only way to keep the queues bounded is to serve a user different from the one served by algorithm A.

These results left open the question of whether there is a stable algorithm for the situation in which $\mathcal{R}$ is finite or the situation in which $\varepsilon > 0$ and $\mathcal{R}^{\inf} > 0$. The paper [11] shows that in both these situations we *can* obtain a stable online algorithm. Let us focus on the case that $\mathcal{R}$ is finite. The algorithm of [11] is called the *Tracking Algorithm* since it operates by trying to track what the adversary's schedule is doing. In order to describe the algorithm in more detail suppose that *after* the online algorithm has made its decision in time slot t, the adversary reveals which user was served by *its* schedule at time step t. Suppose also that the channel rate vector at time t, $(r_0(t), \ldots, r_{n-1}(t)) = (\mu_0, \ldots, \mu_{n-1})$. The *next time* after t that the channel rate vector equals $(\mu_0, \ldots, \mu_{n-1})$ the Tracking algorithm serves the user that the adversary served at time step t. This very simple algorithm ensures that the queue size for each user i is always bounded by $(2F^n + 1)R^{\sup}$, where $F = |\mathcal{R}|$, the number of possible channel rates. Unfortunately, this quantity is exponential in the number of users. Note however that in the Tracking algorithm just described, we need to keep track of which user was last served by the adversary for each possible channel rate vector. We show in [11] that for a slightly more complex version of the Tracking algorithm we can reduce the queuesize bound to $(2nF^2 + 1)R^{\sup}$ by only maintaining state for each possible rate vector for each *pair of users*, rather than for each rate vector for *all* users.

Note that the above description of the Tracking algorithm relies on knowing what the adversary did at the previous time step. In reality we cannot calculate this but we can calculate something similar that is sufficient to make the Tracking algorithm implementable. Recall that for the input process to be schedulable we must have that

$$\sum_{t=t_0}^{t_0+w} a_i(t) \leq (1-\varepsilon) \sum_{t=t_0}^{t_0+w} r_i(t) z_i(t), \tag{4.1}$$

for any t_0. Hence we can approximate the adversary's schedule in the following manner. We divide time into windows of length w. At the end of each window we know exactly what the arrivals were during that window and we know what the channel process was during the window. This means that if can solve the above integer program (4.1) with respect to the variables $z_i(t)$ we can find the adversary's schedule for the previous w time steps. By using ideas similar to the above, this allows the Tracking algorithm to operate. The bound on the queue size becomes $(2nF^2+1)R^{\sup}w$. However, this is still not quite satisfactory since for large values of w, the integer program (4.1) might be intractable. It is shown in [11] however, that the Tracking algorithm is well-defined even if we allow the adversary's schedule to be fractional (i.e. we divide service among multiple different users in each time slot). In this case the integer program becomes a more tractable linear program and we obtain the same bound on queue size.

The above discussion focused on the case in which the feasible rate set is finite. If $\mathcal{R}$ is infinite but $\mathcal{R}^{\inf} > 0$ and $\varepsilon > 0$ we show in [11] that we can still apply these results to obtain a stable schedule by rounding down each channel rate to the closest value $\gamma_k = \mathcal{R}^{\sup}(1-\varepsilon/2)^k$ for $0 \leq k \leq \lceil \log_{1-\varepsilon/2} R^{\sup}/R^{\inf} \rceil$.

4.2. Max-weight produces large queues. We have just described an online scheduling algorithm that is stable whenever $\mathcal{R}$ is finite or when $\varepsilon > 0$ and $\mathcal{R}^{\inf} > 0$. However, the Tracking algorithm is somewhat complex since it requires calculating the adversary's schedule over the past w time steps. It is therefore natural to ask whether there are any simpler algorithms that are also stable whenever these conditions hold. In particular, we would like to know how well the Max-Weight scheduling algorithm of Section 3 performs in this context. Unfortunately, the extremely interesting question of whether Max-Weight is always stable remains unresolved. However, the paper [12] shows that in some cases Max-Weight can perform significantly worse than the Tracking algorithm.

Recall that the more complex version of the Tracking algorithm produces queue sizes that are polynomial in the number of users. In contrast, [12] shows that for the rate set $\mathcal{R}$ used in the EV-DO system the adversary that can create an input process with $\varepsilon = 0$ such that the queue size of one user can be as large as $8184 \cdot 2^n$, when $n < 2048$. The paper [12] also presents some simulation results for a natural example in which $\varepsilon > 0$ and the channel rates are governed by a sequence of users moving past a linear array of basestations. In this case Max-Weight produced queue buildups that were significantly larger than those produced by the Tracking algorithm.

Another contribution of [12] is that it defined a simpler Tracking algorithm that only used state for single users, rather than pairs of users. In particular, for each user i and for each $\mu \in \mathcal{R}$, the simpler Tracking algorithm maintains a counter $c_i(\mu)$ that equals the number of times that the adversary served user i when $r_i(t) = \mu$ minus the number of times that it served user i when $r_i(t) = \mu$. At all times it serves the user with the maximum value of $c_i(r_i(t))$. Although we are not able to prove any bounds for this simpler Tracking algorithm it performed better than the Tracking algorithms with provable performance bounds in many simulation examples.

5. Infinitely backlogged queues - Adversarial channel process. In this section we once again assume that the channel process is generated by an adversary. However, we now consider the case in which each user always has data to serve. Our objective is to maximize $\log R_i$, where R_i is a measure of the service rate to user i. However, in contrast to Section 2, it no longer makes sense to measure a long-term average of service rate using an exponential filter since the feasible service rates could change dramatically over time due to the channel assignment process defined by the adversary. We therefore define $R_i(t)$ to be the total service for user i up to time t. That is, we update $R_i(t)$ by

$$R_i(t+1) = \begin{cases} R_i(t) + r_i(t) & \text{if } i = j \\ R_i(t) & \text{if } i \neq j. \end{cases}$$

In the spirit of competitive analysis, our goal is to define an online scheduling algorithm that always produces a rate assignment such that $\sum_i \log R_i(t)$ is as close as possible to the value produced by the optimal offline algorithm. In [4] it was shown that it is impossible to match the optimal value of the objective. In particular, if there are n users in the system,

THEOREM 5.1. *For any online algorithm A, the adversary can construct a channel rate process such that for some time t, $\sum_i \log R_i(t) \leq \sum_i \log R_i^*(t) - \Omega(n \log n)$, where $R_i^*(t)$ is the total service rate for user i of the offline scheduling algorithm that is optimal for time t.* The proof is short and so we reproduce it here.

Proof. Let p be a parameter. We define a series of special time steps by, $t_k = \sum_i p^k$. We let $T = t_n$. We also define a sequence of sets S_k. For $t_{k-1} < t \leq t_k$ the rate vector is defined by,

$$r_i(t) = 1 \quad \text{if } i \in S_k$$
$$r_i(t) = 0 \quad \text{otherwise.}$$

It remains to define the sets S_k. The initial set $S_0 = \{0, 1, \ldots, n-1\}$. We define,

$$i_k = \arg \min_{i \in S_{k-1}} R_i(t_k).$$

(This means that i_k is the user in S_{k-1} that has received the least amount of service by time t_k). We define $S_k = S_{k-1} - \{i_k\}$. Note that for online algorithms this process is well-defined since i_k does not depend on S_k. We now analyze how much service each user receives. By the definition of i_k,

$$R_{i_k}(t_k) \leq \frac{t_k}{|S_{k-1}|} = \frac{t_k}{n-k+1}.$$

Since $r_{i_k}(t) = 0$ for $t > t_k$,

$$R_{i_k}(T) = R_{i_k}(t_k) \leq \frac{t_k}{n-k+1} = \frac{p^k + p^{k-1} + \ldots + p}{n-k+1}.$$

Hence,

$$\begin{aligned}
\sum_i \log R_i(T) &\leq \log \frac{p}{n} + \log \frac{p^2+p}{n-1} + \ldots + \log(p^n + p^{n-1} + \ldots + p) \\
&= \log \frac{p(p^2+p)\ldots(p^n+p^{n-1}+\ldots+p)}{n!} \\
&\leq \log \frac{p^2 p^3 \ldots p^{k+1} \ldots p^{n+1}}{n!(p-1)^n} \\
&= \left(\sum_{i=1}^{n} \log p^i\right) + n \log \frac{p}{p-1} - \log(n!) \\
&= \left(\sum_{i=1}^{n} \log p^i\right) - \Omega(n \log n).
\end{aligned}$$

(The last equality holds because $\frac{p}{p-1}$ is maximized for $p = 2$.)

On the other hand, there is a valid schedule that assigns all the time slots between t_i and t_{i+1} to user i. This implies,

$$\sum_i \log R_i^*(t) \geq \sum_{i=1}^{n} \log p^i.$$

The result follows. □

We now present a positive result. In particular we show that an extremely simple randomized algorithm can match the bound of Theorem 5.1 up to constant factors.

LEMMA 5.1. *For any sequence of rate vectors, if we serve each user at each time step with probability* $1/n$, *the expected throughputs satisfy*

$$\sum_i \log E[R_i(T)] \geq \sum_i \log R_i^*(T) - O(n \log n).$$

Proof. Follows immediately from the fact that $E[R_i(T)] = \sum_{t \leq T} r_i(t)/n \geq R_i^*(T)/n$. □

We remark that we can approximate the performance of this randomized algorithm by treating it as a fractional schedule that assigns a $1/n$ fraction of each slot to each user and then "tracking" this schedule's performance using the Tracking algorithm of Section 4.

6. Wireless mesh networks. Up until now we have assumed a situation in which we only have a single basestation and multiple mobile receivers. In this section we briefly discuss the case of wireless mesh networks in which traffic is routed through a network consisting of multiple nodes.

We consider the following model. We assume a set of sessions, each one consisting of a path through the network from a source node to a destination node. For each node-pair (i,j) we have a channel rate $r_{i,j}(t)$ that indicates the rate at which we can transmit from node i to node j at time t. At each time step, node i can transmit to at most one neighbor. If it selects neighbor j then the transmission rate is $r_{i,j}(t)$. If session i passes through node i then we let $q_i^k(t)$ be the amount of session-k data queued at node i at time t. We also let n_i^k be the next hop after node i on the path of session k.

For the case in which the channel conditions are generated by a stationary stochastic process and data is injected into each session according to an external arrival process, a generalization of the Max-Weight algorithm defined in Section 3 is known to be stable. For this network version of Max-Weight, at each time step t and at each node i, the scheduler at node i calculates $k^* = \arg\max_k\{(q_i^k(t) - q_{n_i^k}^k(t))r_{i,n_i^k}(t)\}$. If $q_i^{k^*}(t) \geq q_{n_i^{k^*}}^k(t)$ then node i sends session k^* data to node $n_i^{k^*}$ at time t. The amount of data that is sent is $\min\{q_i^{k^*}(t), r_{i,n_i^{k^*}}(t)\}$.

For the case in which the channel conditions are generated by a stationary stochastic process and each session always has data to inject, suppose that we wish to maximize $\sum_k \log R_k(t)$ where $R_k(t)$ is an exponentially filtered average of the session-k data that is served. We can solve this problem by using the Max-Weight scheduling algorithm and then injecting data into session k whenever $\frac{1}{R_k(t)} - \tau q_{begin}^k(t) > 0$, where τ is a small parameter and $q_{begin}^k(t)$ is amount of session-k data queued at the first node on session k's path. This mechanism can be viewed as a method for combining congestion control and scheduling in wireless networks. Note that if a wireless node on the path of session k becomes congested, the Max-Weight scheduling algorithm will create "backpressure" that will cause queue buildups at all nodes on session k's path that are upstream from the congested node. Eventually, $q_{begin}^k(t)$ will become large. When this happens, the rule for data injection means that we are less likely to inject session-k data. Joint optimization of scheduling and congestion control can provide benefits in wireless networks for many reasons. A discussion of this issue may be found in [5].

The above algorithms for scheduling and congestion control in wireless mesh networks are special cases of the Greedy Primal-Dual algorithm for network control defined by Stolyar in [24]. (Similar algorithms were also proposed in [16, 23].) We remark that this Greedy Primal-Dual framework provides numerous extensions to these algorithms, including an algorithm for determining routes through the network in the case that routing is not fixed.

For the case in which the input process is generated by an adversary much less is known. If only the traffic is generated by an adversary but the channel rates are constant then an algorithm similar to Max-Weight can keep all queues stable whenever possible [2]. This result also holds for the variable channel rate case in which all data is destined for a single destination and the packet routes form a rooted tree [13, 14]. However, for the general problem no algorithm is known that keeps the queues stable whenever possible. One partial result that was presented in [10] considers an algorithm that is a hybrid of the Tracking algorithm of Section 4 and the Nearest-to-Source algorithm [6] for wireline networks that always gives priority to data that is closest to its source. The paper [10] shows that this hybrid algorithm is stable as long as the adversary does not correlate the traffic arrivals with the channel rate process. That is, stability holds as long as the adversary does not overload the network if we restrict attention to a set of time slots that all have a common channel rate vector.

7. Discussion. In this survey paper we have presented a number of different models in which scheduling in wireless data networks can be analyzed. In these models we have discussed algorithms that perform well and we have presented some limits on achievable performance. One of the features of these results that we wish to highlight is that different algorithms work well in different models. For example in the stationary channel model Proportional Fair works well when we are trying to provide a fair throughput allocation and each user always has data to serve. In contrast, if the queues are fed by some arrival process then Proportional Fair is not always the ideal algorithm since it can lead to unstable queues.

A number of open problems remain. First as mentioned earlier, what is the performance of the Max-Weight algorithm in adversarial channels? More generally, is there an algorithm that is simpler than the Tracking algorithm and guarantees stability whenever possible in the case of adversarial channels? Lastly, what is the best scheduling algorithm to use in wireless mesh networks? In particular, is there an algorithm that maintains network stability whenever possible and is completely distributed in the sense that it does not require any queue state information to be exchanged between neighboring nodes?

REFERENCES

[1] R. AGRAWAL AND V. SUBRAMANIAN. Optimality of certain channel aware scheduling policies. In *Proceedings of the 40th Annual Allerton Conference on Communication, Control, and Computing*, Monticello, Illinois, October 2002.

[2] W. AIELLO, E. KUSHILEVITZ, R. OSTROVSKY, AND A. ROSEN. Adaptive packet routing for bursty adversarial traffic. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, pp. 359–368, Dallas, TX, May 1998.

[3] M. ANDREWS. Instability of the proportional fair scheduling algorithm for HDR. *IEEE Transactions on Wireless Communications*, 3(5), 2004.

[4] M. ANDREWS. Maximizing profit in overloaded networks. In *Proceedings of IEEE INFOCOM '05*, Miami, FL, March 2005.

[5] M. ANDREWS. Joint optimization of scheduling and congestion control in communication networks. In *Proceedings of 40th Annual Conference on Information Sciences and Systems*, Princeton, NJ, March 2006.

[6] M. ANDREWS, B. AWERBUCH, A. FERNÁNDEZ, J. KLEINBERG, T. LEIGHTON, AND Z. LIU. Universal stability results and performance bounds for greedy contention-resolution protocols. *Journal of the ACM*, 48(1): 39–69, January 2001.

[7] M. ANDREWS, K. KUMARAN, K. RAMANAN, A. STOLYAR, R. VIJAYAKUMAR, AND P. WHITING. CDMA data QoS scheduling on the forward link with variable channel conditions. *Bell Labs Technical Memorandum*, April 2000.

[8] M. ANDREWS, K. KUMARAN, K. RAMANAN, A. STOLYAR, R. VIJAYAKUMAR, AND P. WHITING. Providing quality of service over a shared wireless link. *IEEE Communications Magazine*, February 2001.

[9] M. ANDREWS, L. QIAN, AND A. STOLYAR. Optimal utility based multi-user throughput allocation subject to throughput constraints. In *Proceedings of IEEE INFOCOM '05*, 2005.

[10] M. ANDREWS AND L. ZHANG. Routing and scheduling in multihop wireless networks with time-varying channels. In *Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms*, New Orleans, LA, January 2004.

[11] M. ANDREWS AND L. ZHANG. Scheduling over a time-varying user-dependent channel with applications to high speed wireless data. *Journal of ACM*, September 2005.

[12] M. ANDREWS AND L. ZHANG. Scheduling over non-stationary wireless channels with finite rate sets. *IEEE/ACM Transactions on Networking*, 2006.

[13] E. ANSHELEVICH, D. KEMPE, AND J. KLEINBERG. Stability of load balancing algorithms in dynamic adversarial systems. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*, Montreal, Canada, May 2002.

[14] B. AWERBUCH, P. BERENBRINK, A. BRINKMANN, AND C. SCHEIDELER. Simple routing strategies for adversarial systems. In *Proceedings of the 42nd Annual Symposium on Foundations of Computer Science*, pp. 158–167, Las Vegas, NV, October 2001.

[15] P. BENDER, P. BLACK, M. GROB, R. PADOVANI, AND N. SINDHUSHAYANA A. VITERBI. CDMA/HDR: A bandwidth efficient high speed data service for nomadic users. *IEEE Communications Magazine*, July 2000.

[16] A. ERYILMAZ AND R. SRIKANT. Fair resource allocation in wireless networks using queue-length based scheduling and congestion control. In *Proceedings of IEEE INFOCOM '05*, Miami, FL, March 2005.

[17] J. HOLTZMAN. CDMA forward link waterfilling power control. In *Proceedings of the IEEE Semiannual Vehicular Technology Conference, VTC2000-Spring*, pp. 1663–1667, Tokyo, Japan, May 2000.

[18] A. JALALI, R. PADOVANI, AND R. PANKAJ. Data throughput of CDMA-HDR a high efficiency-high data rate personal communication wireless system. In *Proceedings of the IEEE Semiannual Vehicular Technology Conference, VTC2000-Spring*, Tokyo, Japan, May 2000.

[19] N. KAHALE AND P.E. WRIGHT. Dynamic global packet routing in wireless networks. In *Proceedings of IEEE INFOCOM '97*, Kobe, Japan, April 1997.

[20] H. KUSHNER AND P. WHITING. Asymptotic properties of proportional-fair sharing algorithms. In *40th Annual Allerton Conference on Communication, Control, and Computing*, 2002.

[21] X. LIU, E. CHONG, AND N.B. SHROFF. A framework for opportunistic scheduling in wireless networks. *Computer Networks*, 41(4): 451–474, 2003.

[22] M. NEELY, E. MODIANO, AND C. ROHRS. Power and server allocation in a multi-beam satellite with time varying channels. In *Proceedings of IEEE INFOCOM '02*, New York, NY, June 2002.

[23] M. NEELY, E. MODIANO, AND C. LI. Fairness and optimal stochastic control for heterogeneous networks. In *Proceedings of IEEE INFOCOM '05*, Miami, FL, March 2005.

[24] A. STOLYAR. Maximizing queueing network utility subject to stability: Greedy primal-dual algorithm. *Queueing Systems*, 50(4): 401–457, 2005.

[25] A. STOLYAR. On the asymptotic optimality of the gradient scheduling algorithm for multiuser throughput allocation. *Operations Research*, 53: 12–25, 2005.

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

[27] L. TASSIULAS AND A. EPHREMIDES. Dynamic server allocation to parallel queues with randomly varying connectivity. *IEEE Transactions on Information Theory*, 30: 466–478, 1993.

[28] D. TSE. Multiuser diversity in wireless networks. `http://www.eecs.berkeley.edu/~dtse/stanford416.ps`.

WIRELESS CHANNEL PARAMETERS MAXIMIZING TCP THROUGHPUT

FRANÇOIS BACCELLI*, RENE L. CRUZ†, AND ANTONIO NUCCI‡

Abstract. We consider a single TCP session traversing a wireless channel, with a constant signal to interference and noise ratio (SINR) at the receiver. We consider the problem of determining the optimal transmission energy per bit, to maximize TCP throughput. Specifically, in the case where direct sequence spread spectrum modulation is used over a fixed bandwidth channel, we find the optimal processing gain m that maximizes TCP throughput. In the case where there is a high signal to noise ratio, we consider the scenario where adaptive modulation is used over a fixed bandwidth channel, and find the optimal symbol alphabet size M to maximize TCP throughput. Block codes applied to each packet for forward error correction can also be used, and in that case we consider the joint optimization of the coding rate to maximize TCP throughput. Finally, we discuss the issue of assigning target SINR values. In order to carry out our analysis, we obtain a TCP throughput formula in terms of the packet transmission error probability p and the transmission capacity C, which is of independent interest. In our TCP model, the window size is cut in half for each packet transmission loss, and also cut in half whenever the window size exceeds C. This formula is then used to characterize the optimal processing gain or the optimal symbol alphabet size as the solution of a simple fixed point equation that depends on the wireless channel parameters and the parameters of the TCP connection.

Key words. CDMA, adaptive modulation, processing gain, block coding, signal to noise and interference ratio, power control, bandwidth sharing, congestion control, congestion avoidance, additive increase multiplicative decrease algorithm, TCP throughput, optimization, stochastic process, stationary distribution, Mellin transform.

AMS(MOS) subject classifications. Primary 94A05, 94C99, 60K30.

1. Introduction.

Cellular wireless networks were originally designed to support voice, which has stringent delay requirements. In these networks, a power control algorithm is used to maintain a target SINR for each user. The power control algorithm adapts to fast multi-path fading that arises due to mobility of users or the sources of scattering, so that a constant bit rate and a required maximum bit error rate is maintained for each connection, with low transport latency. Thus, for example, when a user encounters a fading channel condition, the transmission power is boosted so that the voice conversation can continue in real time.

These systems have been adapted to carry data as well. A fixed capacity channel may be allocated for a data user. We are generally interested in optimizing the channel parameters in order to provide the best performance for the data user. For simplicity here we assume that a data user corresponds to a single TCP connection. We assume that the channel is

*INRIA-ENS (francois.baccelli@ens.fr).

†UCSD, USA (rcruz@ucsd.edu).

‡This author was with Sprint ATL, USA, when this work was initiated. He is now with Narus, USA (anucci@narus.com).

allocated a SINR which will be maintained to a constant target value. We focus on the case of a long lived TCP connection. The target SINR value may be adapted over time to respond to user mobility, and we consider the regime where the TCP connection reaches a steady state between updates of the target SINR value.

We consider first the case of CDMA, and consider the problem of optimizing the processing gain m and coding rate ρ to maximize TCP throughput. By adjusting the processing gain and coding rate, we trade-off the bit transmission rate C and packet transmission error rate p. By making the processing gain m large we increase the energy per bit and so the packet transmission loss probability p is small. However, the bit transmission rate C is proportional to $1/m$.

We also consider the context of a high SINR channel over a fixed bandiwidth. In this case, we study adaptive modulation, where the symbol alphabet size M is adapted. In this case the raw transmission rate C is proportional to $\log_2(M)$, but the probability of a packet error p grows quickly with M. We find the optimal value of the alphabet size M in order to maximize TCP throughput.

To understand the nature of the optimization we consider, it is useful to think of two extremes. At one extreme we can make the bit-transmission rate C high, but the packet transmission error rate p will be large. Packet transmission errors will generally cause the TCP protocol to reduce its window size, and in turn decrease throughput. At the other extreme we can make the packet transmission error rate p very small, at the expense of reduced transmission rate C. The bit transmission rate C ultimately limits the TCP window size, since buffer overflows will occur when the window gets sufficiently large, and TCP will cut the window size in half, responding to the congestion that occurs at the buffer. Thus, TCP throughput is small at this extreme as well.

In order to find the optimal operating point, we consider a model for analyzing the TCP throughput where packet losses due to transmission errors and packet losses due to congestion events are distinguished. In Section 3, we consider a fluid model where the window size is cut in half for each packet loss due to a transmission error or buffer overflow event. The packet loss probability is p, and we assume that a congestion event happens when the window size reaches a value that matches the bit-transmission rate, C, on the channel. This model is appropriate for small buffers. This makes sense within e.g. the CDMA context where the bit-transmission rate is small, compared to what is available on wireline networks, and where large downlink buffers would imply large RTTs and hence poor TCP performance as TCP throughput is known to be roughly inversely proportional to RTT.

For this model, we find a formula for the TCP throughput as a function of p and C, which is of independent interest. This is a generalization of the well known"square root" formula (e.g. see [11]) for TCP throughput

where packet transmission errors and buffer overflows are not separately modeled. We also obtain formulas for the probability density function of the TCP window size.

In Section 4, we show how to use the analytical framework for TCP throughput in the particular case of wireless channels with fixed SINR. We consider several cases ranging from CDMA to adaptive modulation, with or without FEC. We show that the TCP throughput can be fairly sensitive to the physical layer parameters, i.e. the processing gain m or symbol alphabet size M, and we give simple analytical characterizations of the optimal values for these parameters.

We also discuss, in Section 5, the problem of assignment of target SINR values in a network context, and specifically discuss the case of a wireless cellular CDMA downlink.

Before describing our model and analysis in more detail, we first discuss some related work.

2. Related work. Many authors have considered the general problem of performance of the TCP protocol over wireless links. A comparison of various approaches to the problem is given in [4].

One branch of related work is concerned with the problem of determining whether packet losses are due to congestion or due to transmission errors, so that TCP can go into congestion avoidance mode only when packet losses are due to congestion. A second branch of work considers the approach of splitting the TCP connection at the wire-line/wireless infrastructure boundary, so that the TCP connection is isolated from the packet transmission errors on the wireless channel. A third approach to optimizing TCP performance over wireless channels is to optimize the link layer for TCP performance.

The approach we take in this paper falls into the third category.

Zorzi and Rao [14] considered the effect of correlated errors on TCP throughput. Correlated errors typically occur due to multi-path fading. In our model, we assume a fixed SINR (thanks to power control), so errors can be modeled as independent.

Chaskar, Lakshman, and Madhow [5] consider the use of link layer ARQ over the wireless channel to hide packet transmission errors. In this case, TCP primarily reacts to buffer overflows only. In [5], it is suggested that the physical layer parameters should be optimized so that the buffer overflow probability q times the bandwidth delay product squared is equal to one.

Liu, Goeckel, and Towsley [10] considered the problem of adapting the coding rate ρ to the channel conditions, with the objective of maximizing TCP throughput. It was already noted in [10] that optimal values of operating parameters for the channel for TCP were different than those for UDP.

In [9], Liu, Zhou and Giannakis used simulation to study cross-layer optimization within the adaptive modulation setting.

The main novelty of the present paper is the fact that it provides an analytic framework for this adaptation.

Several recent papers provide analytical formulas for the throughput of a large collection of competing TCP flows with both congestion and transmission error losses (see [2] and the references therein). The main difference with [2] and related papers is that we are here focusing on the rate of a *single* TCP connection constrained to remain *below the rate* C and subject to both random transmission error losses and to losses that occur when its rate reaches or exceeds C.

To the best of our knowledge, this setting, the associated formulas for TCP throughput and the analytic characterizations of the optimal wireless channel parameters that are derived in the present paper are all new.

3. TCP throughput.

3.1. Model. In this section, we analyze the throughput of a single TCP flow over a wireless channel. Each packet contains L bits, and when it is transmitted over the channel, it is lost with probability p. We assume that packet transmission errors are independent. Later on, in Section 5, we will consider how the packet transmission loss probability p depends on γ and other physical layer parameters. We assume it takes L/C seconds to transmit each packet on the channel. The parameter C is called the raw transmission capacity and has units of bits/sec.

We now consider a fluid model for the TCP flow. Our model is in terms of the parameters C and p discussed above. Consider a random process $X(t)$ that models the instantaneous throughput for the TCP flow. The instantaneous throughput is assumed to be proportional to the TCP window size. Our model for the dynamics of $X(t)$ is as follows. Let R be the round trip delay (assumed constant here), in units of seconds. When there is no loss, at time t, $X(t)$ increases at rate L/R^2. If $X(t)$ reaches C, a congestion event occurs, causing $X(t)$ to be reduced in half to the value $C/2$. Packet losses are modeled by a time in-homogeneous Poisson process, where the rate of loss at time t is $\lambda(t) = pX(t)/L$. This reflects the fact that the rate of packet loss is higher when the packet rate is higher. When there is a packet loss at time t, the value of $X(t)$ is reduced in half.

Of course a more refined model would take into account the presence of a buffer and the fact that losses only take place when this buffer overflows. In Section 6, we will show by simulation that the conclusions obtained from our simplified bufferless model are still valid for this refined model.

3.2. Distributions. We first define some notation. Let $\alpha = pR^2/L^2$. Define $\Lambda_0 = 1$ and for $l \geq 1$ define

$$\Lambda_l = \prod_{j=1}^{l} \left(\frac{-4}{2^{2j} - 1} \right) . \tag{3.1}$$

For $l \geq 1$ and $n \geq 2$ define

$$a_{l,n} = \Lambda_{l-1} \exp(-(2^{2(l-1)} - 1)\alpha(C/2^n)^2/2) \left[1 + (\frac{4}{2^{2l} - 1}) \exp(-(3)2^{2(l-1)}\alpha(C/2^n)^2/2)\right] . \tag{3.2}$$

The following is proved in Appendix B.

THEOREM 3.1. *Let $f(x)$ be the stationary probability density function for the instantaneous throughput $X(t)$ of the TCP connection. The density f satisfies the differential equation*

$$\frac{df(x)}{dx} = \begin{cases} -\alpha x f(x) & \text{if } C/2 < x \leq C \\ -\alpha x f(x) + 4\alpha x f(2x) & \text{if } 0 \leq x < C/2 \end{cases} , \tag{3.3}$$

where $\alpha = pR^2/L^2$. The density $f(x)$ is discontinuous at $x = C/2$ and such that

$$f((C/2)^+) - f((C/2)^-) = f(C^-) . \tag{3.4}$$

The density $f(x)$ is given by

$$f(x) = \sum_{l=0}^{n} V_{n-l}\Lambda_l e^{-2^{2l}\alpha x^2/2} \tag{3.5}$$

if $C/2^{n+1} < x < C/2^n$ for $n \geq 0$, 0 otherwise, where the constants $V_0, V_1, \ldots$ are the solutions to the equations

$$V_1 = V_0(1 + (1/3)e^{-3\alpha C^2/8}) , \tag{3.6}$$

$$V_n = \sum_{l=1}^{n} a_{l,n}V_{n-l} \text{ for } n > 1 , \tag{3.7}$$

and

$$\sum_{m=0}^{\infty} V_m \left[\int_{C/2^{m+1}}^{C/2^m} \exp(-\alpha x^2/2)dx\right] = \left[\sum_{k=0}^{\infty} 2^{-k}\Lambda_k\right]^{-1} . \tag{3.8}$$

It is interesting to note that in all cases examined so far, the constants V_n appear to converge to a constant as $n \to \infty$.

3.3. Limiting cases. First consider the case where $p = \alpha = 0$. In this case it can be verified that $\sum_{l=0}^{n} V_{n-l}\Lambda_l = 0$ for all $n > 1$ and hence $f(x)$ is a uniform distribution on $[C/2, C]$. This is expected since in this case the trajectory of $X(t)$ is a sawtooth pattern varying between $C/2$ and C.