Time-Varying Fair Queueing Scheduling for Multicode CDMA Based on Dynamic Programming

Anastasios Stamoulis, Nicholas D. Sidiropoulos, *Senior Member, IEEE*, and Georgios B. Giannakis, *Fellow, IEEE*

***Abstract*—Fair queueing (FQ) algorithms, which have been proposed for quality of service (QoS) wireline/wireless networking, rely on the fundamental idea that the service rate allocated to each user is proportional to a positive weight. Targeting wireless data networks with a multicode CDMA-based physical layer, we develop FQ with time-varying weight assignment in order to minimize the queueing delays of mobile users. Applying dynamic programming, we design a computationally efficient algorithm which produces the optimal service rates while obeying 1) constraints imposed by the underlying physical layer and 2) QoS requirements. Furthermore, we study how information about the underlying channel quality can be incorporated into the scheduler to improve network performance. Simulations illustrate the merits of our designs.**

***Index Terms*—Code-division multiple access (CDMA), dynamic programming, fair queueing, quality of service, scheduling.**

I. Introduction

In INTEGRATED services networks, the provision of quality of service (QoS) guarantees to individual sessions depends critically upon the scheduling algorithm employed at the network switches. In wireless networks, the scheduler (which resides at the base station in a centralized implementation) allocates bandwidth by, e.g., assigning slots (in time-division multiple access—TDMA—environments) or codes (in code-division multiple access—CDMA—environments). Though in second-generation wireless networks, the scheduler needs to allocate bandwidth only for voice and low-rate data traffic, it is expected that in third-generation broad-band wireless networks, a plethora of applications with diverse QoS requirements will need to be supported.

In both wireline and wireless networks, the generalized processor sharing (GPS) [1] discipline and the numerous fair queueing (FQ) algorithms are widely considered as the primary scheduler candidates, as GPS has been shown to provide both minimum service rate guarantees and isolation from ill-behaved traffic sources. Not only have GPS-based algorithms been implemented in actual gigabit switches in wired networks, but also they have been studied in the context of the emerging broad-band wireless networks (see, e.g., [2], [3], and references therein). The fundamental notion in GPS-based algorithms is that the amount of service session m receives from the switch (in terms of transmitted packets) is proportional to a positive weight β_m. As a result, GPS (and its numerous FQ variants) is capable of delivering bandwidth guarantees; the latter translate to delay guarantees as long as there is an upper bound on the amount of incoming traffic (this bound could be either deterministic or stochastic).

One of the major shortcomings of GPS is that the service guarantees provided to session m are controlled by just one parameter, the weight β_m. Hence, the *delay-bandwidth coupling*, which refers to the mutual dependence between delay and throughput guarantees (i.e., in order to guarantee small delays, a large portion of the bandwidth should be reserved). To appreciate why the delay-bandwidth coupling is a shortcoming, one needs to take into consideration that future networks will support multirate multimedia services with widely diverse delay and bandwidth specifications. For example, video and audio have delay requirements of the same order, but video has an order of magnitude greater bandwidth requirement than audio. Therefore, delay-bandwidth coupling could lead to bandwidth underutilization.

In this paper, we look at the problem of minimizing queueing delays in wireless networks which employ FQ (as the bandwidth allocation policy), and multicode CDMA (as the physical layer transmission/reception technique). We base our approach on a time-varying weight assignment, which dispenses with the delay-bandwidth coupling, while still obeying QoS requirements (in terms of minimum guaranteed bandwidth to individual sessions). Using dynamic programming (DP), we design a computationally efficient algorithm, which produces the optimal weights β_m's, that minimize a cost function representing the queueing delays of the mobile users. Unlike existing work, our algorithm takes into explicit account the discrete nature of the service rates (as they are provided by the underlying physical layer), and, as a matter of fact, capitalizes on this discrete nature to reduce computational complexity.

Furthermore, we investigate how information about channel quality can be incorporated into the scheduler. Intuitively, a user with a "good" channel, i.e., with sufficiently high signal-to-noise-ratio (SNR), should be allocated a fair number of CDMA codes to maximize throughput (while channel conditions remain favorable). On the other hand, whenever the channel is "bad," the user should be discouraged from transmitting data packets. Existing work has mainly addressed the extreme case where only one user is allowed to transmit (see, e.g., [4]–[7] and

Manuscript received February 20, 2002; revised July 29, 2002 and November 9, 2002; accepted November 13, 2002. The editor coordinating the review of this paper and approving it for publication is M. Zorzi. The work of N. Sidiropoulos was supported by NSF/Wireless CCR-9979295. This work was presented in part at the IEEE Globecom, San Antonio, TX, November 2001.

A. Stamoulis was with the Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis, MN 55455 USA. He is now with Qualcomm, Inc., San Diego, CA 92121 USA (anastasios.stamoulis@qualcomm.com).

N. D. Sidiropoulos and G. B. Giannakis are with the Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis, MN 55455 USA (nikos@ece.umn.edu; georgios@ece.umn.edu).

Digital Object Identifier 10.1109/TWC.2003.821151

references therein). Our approach is more general, because we incorporate channel quality as a weighting factor in a properly defined cost function. Our framework allows the simultaneous transmission by more than one users (as long as they have sufficiently "good" channels), and has the potential to lead to improved QoS per user.

The rest of this paper is structured as follows. Section II states the problem and describes our modeling assumptions. Section III presents our algorithm and explains why our bandwidth allocation policy is optimal (given our operating assumptions). Section IV focuses on how channel quality can be incorporated into the bandwidth allocation process. In both sections, the merits of our approach are illustrated through simulations. Conclusions are drawn in Section V.

II. Model Description and Problem Statement

We focus on a single cell in a wireless multicode CDMA network, where mobile users receive service from the basestation. The basestation allocates bandwidth to mobile users using a fair queueing algorithm, decides on the corresponding CDMA codes, and communicates bandwidth/code assignments to mobile users using a demand-assignment medium access control (MAC) protocol. Next, we provide a brief description of the scheduler, MAC, and the underlying multicode CDMA, and then we state the problem we endeavor to solve.

A. Fair Queueing Scheduler

The bandwidth allocation policy is based on the GPS scheduling algorithm [1] also known as weighted fair queueing [8]. From a network-wide point of view, GPS efficiently utilizes the available resources as it facilitates statistical multiplexing. From a user perspective, GPS guarantees to the sessions that 1) network resources are allocated irrespective of the behavior of the other sessions (which refers to the *isolation* property of the scheduler) and 2) whenever network resources become available (e.g., in underloaded scenarios), the extra resources are distributed to active sessions (the *fairness* property of the scheduler).

According to [1], a GPS server operates at a fixed rate r and is work conserving, i.e., the server is not idle if there are backlogged packets to be transmitted. The scheduler usually resides at the output links of a network switch [9]. Each session m is characterized by a positive constant β_m, and the amount of service $R_m(\tau,t)$ session m receives in the interval $(\tau,t]$ is proportional to β_m, provided that the session is continuously backlogged. Formally, under GPS, if session m is continuously backlogged in $(\tau,t]$, then it holds that

$$\frac{R_m(\tau,t)}{R_\mu(\tau,t)} \geq \frac{\beta_m}{\beta_\mu}$$

for all sessions μ that have also received some service in this time interval. It follows that in the worst case, the minimum guaranteed rate r_m given to session m is $r_m = r\beta_m/\sum_{\mu=1}^{M}\beta_\mu$, where M is the maximum number of sessions that could be active in the system. Therefore, a lower bound for the amount of service that session m is guaranteed is: $R_m^l(\tau,t) := (t-\tau)r\beta_m/\sum_{\mu=1}^{M}\beta\mu$. If session m is (σ_m,ρ_m)-leaky bucket constrained,[1] and the minimum guaranteed rate is such that $r_m \geq \rho_m$, then the maximum delay is $D_m^{\max} \leq \sigma_m/r_m$ (note that this bound could be loose [1]).

Effectively, GPS offers perfect isolation, because every session is guaranteed its portion of the bandwidth irrespective of the behavior of the other sessions. From this point of view, GPS is reminiscent of fixed-assignment TDMA or frequency-division multiple access (FDMA) physical layer multiplexing techniques. What is radically different about GPS is its perfect fairness property.[2]Whenever a session m generates traffic at a rate less than r_m, then the "extra" bandwidth is allocated to other sessions proportionally to their respective weights. Let us clarify the operation of GPS in a wireless network using the following simple example: suppose that in a pico-cell three mobile users are assigned to the base station. One (high-rate) user has a weight of $\beta_{hr} = 1$, and the two (low-rate) users have a weight of $\beta_{lr} = 0.5$. When all users are active, the high-rate user will take 50% of the bandwidth, and each of the low-rate users 25%. If one of the low-rate users becomes silent, then the extra 25% of the bandwidth will be allocated to the other users: the high-rate will have now 66%, and the low-rate 34%. Note that the extra bandwidth can be used in multiple ways, for example, to increase the information rate, or to decrease the transmitted power through the use of a more powerful channel code.

GPS belongs to the family of rate-based schedulers [12], which attempt to provide bandwidth guarantees to sessions (note that bandwidth guarantees yield delay guarantees if description of the incoming traffic is available). When the sessions have a nominal, long-term average rate (the *sustainable cell rate* or SCR in ATM terminology), then the allocation of β's appears to be straightforward. The situation becomes more complicated if we consider that network traffic could be bursty or self-similar. Though there has been work on the weight assignment problem (see, e.g., [13]), it is still considered quite challenging (see, e.g., [14]). One of the contributions of this work is that our algorithm yields the optimal set of weights which minimizes a cost function representing the queueing delays of mobile users. Before we give a mathematical description of this cost function in Section II-C, let us cast an eye to how bandwidth is actually allocated at the physical layer.

B. Bandwidth Allocation Under Multicode CDMA

At the physical layer, we assume a multicode CDMA transmission/reception scheme. There are C available codes[3] (these codes could be, e.g., Pseudo-Noise or Walsh–Hadamard), which can be allocated to mobile users. Each user m is allocated ϕ_m codes, and splits the information stream into ϕ_m substreams which are transmitted simultaneously using each of the ϕ_m codes: it readily follows that if user m has Q_m data symbols

[1]For any interval $(\tau,t]$, the traffic that is generated is upper bounded by $\sigma_m + \rho_m(t-\tau)$ [10].

[2]Formally, if two sessions m, μ, are continuously backlogged in $(\tau,t]$, then GPS postulates that $(R_m(\tau,t)/R_\mu(\tau,t)) = (\beta_m/\beta_\mu)$. Since GPS can only be approximated in practice by an FQ algorithm, the accuracy of the approximation may be judged by the difference $|(R_m(\tau,t)/R_\mu(\tau,t)) = (\beta_m/\beta_\mu)|$. Discussion and references related to fairness may be found in, e.g., [11].

[3]Note that the capacity C is "soft" as it depends on channel conditions, power control, etc.

to transmit, then Q_m/ϕ_m yields a measure of the time it takes to transmit them. Herein we assume that channel conditions, power control, and the detection mechanism at the basestation allow the successful use of all C codes: $\gamma_{m,c} > \gamma$, where γ is an SNR threshold for a prescribed bit-error-rate (BER) probability, and $\gamma_{m,c}$ is the cth code of the mth user. Though, in general, CDMA is an interference-limited environment, crosstalk can be kept at modest levels in narrow-band, quasi-synchronous systems with the aid of power control and forward error correction. Hence, it can be assumed that all codes are available (both in the uplink, and the downlink scenario). Furthermore, in wide-band CDMA, there exist judiciously designed CDMA codes (see [15], [16], and references therein) which guarantee elimination of crosstalk irrespective of the possibly unknown frequency selectivity (under mild assumptions on delay spread and time synchronization among users). Chip interleaving, for example, is a particularly simple way to turn interchip interference (ICI), which destroys the orthogonality among the codes, into intersymbol interference (ISI), which can be taken care of by equalization, thus maintaining *exact* code orthogonality, through simple periodic chip interleaving.

Based on the dynamic range of SNR values $\gamma_{m,c}$ we can differentiate between two cases.

Case I: Power control and power budget of each mobile user are such that all SNR values $\gamma_{m,c}$ are approximately equal. This could be accomplished with very fast power control, accurate SNR estimation, and an algorithm such as in [17]. In this case, all users experience "good" channels.

Case II: Power control is not fast enough to track channel fluctuations, and, consequently, the SNR values $\gamma_{m,c}$'s have disparate values per user (i.e., $\gamma_{m,c}$ can be disparate as function of m, but not as a function of c for a given m). Such a scenario could arise in cases where: 1) the power control can track the slow-time fading, but not the fast time-scale fading or 2) a mobile user experiences a deep fade, and the power budget is not enough to compensate for it. Recall that the signal power p_t which impinges on the receive antenna of a mobile user is given at any time t as the product of two independent stochastic processes, i.e., $p_t = |h_t|^2 s_t$, where s_t is the slow-fading process due to path loss and shadowing effects, and $|h_t|^2$ is the fast variation due to scattering [18]).

Both cases can be handled by our DP-based algorithm. The key of this uniform treatment lies in the proper definition of the cost function, and the weights associated with the QoS provided to each user. For Case I, the weights provide different QoS priorities. For Case II, in Section IV we will see how the weights (which depend on the different SNR values) provide a form of multiuser diversity, and can actually be used to improve the overall network performance.

Before we present this cost function, two comments are due.

Comment I: Unlike wired networks, the implementation of GPS in multicode CDMA networks appears to be straightforward.[4] Given the assumption on the successful use of all C codes, ϕ_m/C essentially denotes the bandwidth which is allocated to user m, and GPS is implemented by setting

$$\phi_m = \left\lfloor \frac{\beta_m}{\sum_{\mu\in\mathcal{A}} \beta_\mu} C \right\rfloor \tag{1}$$

where $\mathcal{A}$ is the set of active users (note that with C sufficiently large and frequent code reassignments, the approximation error in implementing GPS using (1) can be made small). Our DP algorithm produces the optimal code allocation $\{\phi_m\}_{m=1}^M$, which implicitly yields the optimal weight allocation $\{\beta_m\}_{m=1}^M$.

Comment II: The network and the physical layer can be tied together using a two-phase demand-assignment MAC protocol. During the first phase, each user m notifies the base station about its intention to transmit (and the queue length for reasons we will explain later); the base station calculates the ϕ_m's, and notifies each user about the corresponding code assignment. During the second phase, users rely on these codes to transmit (at possibly different rates). Note that: 1) the duration of the reservation phase can be reduced if users piggy-back their queue-lengths in prespecified intervals) and 2) the overall scheme becomes much simpler in the downlink case, as the basestation is aware of the queue lengths of all data streams. Demand-assignment MAC protocols constitute a well-studied field, and we will not further elaborate on them (see, e.g., [19] for details).

C. Problem Statement

Our objective is to come up with the solution $(\phi_1, \ldots, \phi_M)$ of the problem

$$\text{minimize} \sum_{m=1}^{M} \frac{Q_m}{w_m \phi_m} \text{ subject to} \tag{2}$$

$$\sum_{m=1}^{M} \phi_m = C, \quad 1 \le L_m \le \phi_m \le U_m \le C, \quad \phi_m \in \mathbb{N}. \tag{3}$$

In other words, we want to calculate the number of codes that are to be allocated to each user so that a cost function representing the queueing delays is minimized.[5] In particular, $(1/M)\sum_{m=1}^{M}\lceil Q_m/\phi_m\rceil$ is the average "evacuation delay" [20] if no more packet arrivals occur. When all users experience good channels, the constants $w_m > 0, 1 \le m \le M$, allow us to introduce different priorities in the system: low (high) priority users should be assigned constants $w_{lp}(w_{hp})$ with $w_{lp} > w_{hp}$. Moreover, as we discuss in Section IV, they can be used to incorporate information about channel quality (in this case, users with good channels are assigned smaller weights).

The constants $\{L_m, U_m\}_{m=1}^M$ are integers indicating lower and upper bounds respectively on the number of codes that are to be allocated to session m. Note that $\{L_m, U_m\}_{m=1}^M$ enable us to impose QoS constraints on the set of feasible solutions: on the one hand, L_m yields a minimum throughput guarantee, and, on the other hand, U_m assures that a greedy (or malicious) source will not be allocated a large portion of

[4]In wired networks, GPS assumes a fluid model of traffic. Hence, in practice GPS needs to be approximated by a packet fair queueing (PFQ) algorithm [1]. Starting with weighted fair queueing (WFQ) [8], there has been a lot of work on approximating GPS (see, e.g., [3] and references therein).

[5]In general, the constraint should be $\sum_{m=1}^{M}\phi_m \le C$; however, we assume that the constraints $\{U_m\}_{m=1}^M$ are such that all codes can be allocated. Hence, the equality in the constraint $\sum_{m=1}^{M}\phi_m = C$. We thank the anonymous reviewers for pointing this out.

the bandwidth. Furthermore, $\{L_m, U_m\}_{m=1}^M$ may be used to control the tradeoff between latency and average throughput, as discussed in Section III-E. Also note that for certain choices of $\{L_m, U_m\}_{m=1}^M$, C the problem may be infeasible or trivial. This can be easily detected in a preprocessing step. For brevity, we assume meaningful choices of $\{L_m, U_m\}_{m=1}^M$, C throughout, leading to problems that admit multiple feasible solutions, for which we seek the optimum in the sense of minimizing the cost in (2).

Before we proceed with the presentation of our DP-based solution, a technical remark is due at this point. Suppose that after a time instant, no more packets are allowed to be inserted to the queues. If codes are not reassigned during the "evacuation" interval, the average evacuation delay is approximately equal to $(1/M)\sum_{m=1}^M Q_m/\phi_m$. However, as transmissions are allowed to overlap, the maximum delay in the system is upper bounded by $\max_{1\leq m\leq M}\{\lceil Q_m/\phi_m\rceil\}$. In fact, the delay can be made much smaller than $\max_{1\leq m\leq M}\{\lceil Q_m/\phi_m\rceil\}$, because when the queue of the μth user drains out, the μth user's codes can be allocated to the other active users. Hence, the cost function $\sum_{m=1}^M Q_m/\phi_m$ serves as a pessimistic estimate of the delay, *but* it leads to a tractable algorithmic development, as we describe next.

III. Dynamic-Programming-Based Solution

A. Preliminary

In general, DP [21] can be used to search for the M-tuple $\{x_m\}_{m=1}^M$ of finite-alphabet "state" variables that minimizes $\sum_{m=1}^M \mathrm{cost}_m(x_m, x_{m-1})$, where x_0 is given and $\mathrm{cost}_m(\cdot,\cdot)$ is some arbitrary "one-step transition" cost. The $\sum_{m=1}^M$ can be the usual arithmetic sum, or, e.g., the $\max_{m=1}^M(\cdot)$: hence DP can be used to minimize $\max_{m=1}^M \mathrm{cost}(x_m, x_{m-1})$; this follows from the principle of optimality (a.k.a. principle of contradiction) of DP [21]. DP (with the Viterbi algorithm as a well-known incarnation) avoids exhaustive search and makes it possible to find an optimum solution in time linear in M: assuming that calculating $\mathrm{cost}_m(\cdot,\cdot)$ is $O(1)$, and the size of the finite alphabet is A, the complexity of DP is $O(A^2M)$ or less, depending on the specific cost structure. To see why DP applies to the problem at hand, define

$$x_m := \sum_{l=1}^m \phi_l$$

and note that $\phi_m = x_m - x_{m-1}$, hence

$$\sum_{m=1}^M \frac{Q_m}{w_m\phi_m} = \sum_{m=1}^M \frac{Q_m}{w_m}\frac{1}{x_m - x_{m-1}}.$$

By optimality, the best terminal state is $x_M = C$, and specifying $\mathrm{cost}_m(x_m, x_{m-1}) = \infty$ whenever $x_m \leq x_{m-1}$. The inequality constraints in (3) can be enforced in a similar fashion by restricting the state fan-out. Finally, for all allowable state transitions we set $\mathrm{cost}_m(x_m, x_{m-1}) = (Q_m/w_m)(1/(x_m - x_{m-1}))$.

B. An $O(C^2M)$ Algorithm

For the moment, let us ignore the constants $\{L_m, U_m\}_{m=1}^M$. Consider Fig. 1, which depicts nodes at stage i and their respective potential predecessors at stage $(i-1)$. There are M stages, and each stage has C nodes. At stage i, the node tags correspond to all possible values of $\sum_{m=1}^i \phi_m$, and similarly for stage $(i-1)$. Hence, the node tags are 1, 2, ..., C. The cost associated with the transition from node with tag $n^{(i-1)}$ at stage $(i-1)$ to node with tag $n^{(i)}$ at stage i is $(Q_i/w_i)(1/(n^{(i)} - n^{(i-1)}))$.

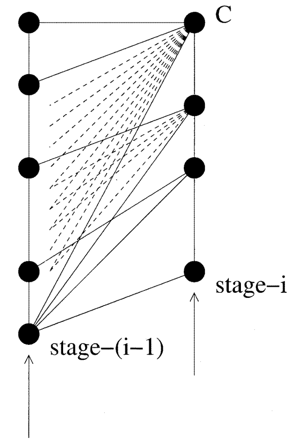

Fig. 1. Nodes (states) at stage i and predecessors at stage $(i-1)$.

At stage i, the bottom node has just one predecessor which is one node apart (remember that each user is to be allocated at least one code). The next node going upwards has just two predecessors, the furthest of which is two nodes apart. If we take $\{L_m, U_m\}_{m=1}^M$ into account, then some transition steps are eliminated. In particular: 1) node n at stage $(i-1)$ is not allowed to lead to nodes $1, \ldots, n + L_i - 1$ of stage i and 2) node n at stage i has up to U_i predecessors, the furthest of which is $n - U_i$ apart.

Each node at stage i is visited in turn, and a decision is made as to which of the associated potential predecessors is best for the node at hand. To do this, we need to calculate the transition cost, add the respective results to the corresponding cumulative costs of the potential predecessor nodes, pick the one that gives minimum error, update the cumulative cost of the node at hand, set up a pointer to its best predecessor, then move on to the next node, the next stage, and so on.

With no constraints (i.e., $L_m = 1$, $U_m = C$, $\forall m$), the computational complexity of the aforementioned algorithm is $O(C^2M)$. With nontrivial QoS constraints, the execution time of the algorithm is decreased as transitions in the trellis diagram are expurgated. In any case, the computational burden falls within the capabilities of modern DSP processors. For example, consider a system with $C = 128$ codes, and $M = 32$ users, where the code reallocation is performed every 16.7 ms (this could correspond, for example, in ten slots in the HDR/CDMA system; each slot in HDR/CDMA is 1.67 ms long). Then, our DP algorithm would require in the order of 31 MIPS (note that modern programmable DSP processors deliver more than 100 MIPS).

C. An $O(UM^2)$ Algorithm

An alternative DP solution is based on the intuition that more codes should be allocated to users with high Q/w's. Such an algorithm would proceed as follows. First, the

Q_m/w_m's are sorted in descending order with computational cost $O(M \log M)$. To simplify notation, we will assume that $(Q_1/w_1) \geq (Q_2/w_2) \geq \cdots \geq (Q_M/w_M)$, and we observe that the code allocation procedure should result in ϕ's that have descending order: $\phi_1 \geq \phi_2 \cdots \geq \phi_M$ (this is because we want to minimize $\sum_{m=1}^{M}(Q_m/w_m\phi_m)$). Instead of evaluating directly the values of $\phi_1, \ldots, \phi_M$, one could evaluate the breakpoints where the ϕ's are decreased. For example, suppose that $M = 3, C = 5$, and the code allocation is $\phi_1 = \phi_2 = 2$, $\phi_3 = 1$. The breakpoint is 3, because at the third node we move from the allocation "two codes per user" to the allocation "one code per user". Overall, this approach pays off when C is relatively large with respect to M, and $L_m = 1, U_m = U$, for all m. Then, the cost of the overall algorithm is $O(UM^2)$ (which subsumes the $O(M \log M)$ cost of the sorting algorithm).

Formally, following the material in [22], we can do DP over breakpoint variables b_i. We start by writing the cost as

$$\sum_{i=1}^{U-1} g(b_i, b_{i-1})$$

where b_1 is the point that the DP trellis switches from allocation U to allocation U-1, b_2 is the point where the DP trellis switches from allocation U-1 to allocation U-2, and so on. Breakpoints assume the "parking" value of $M+1$ if some switches are not needed (i.e., C is large enough). Each breakpoint variable takes values in $1, \ldots, M$. Hence, we have M states per stage and $U-1$ stages; for each stage we need to look back at $O(M)$ states in the previous stage. This leads to complexity $O(UM^2)$ because the computation of all $g(b_i, b_{i-1})$ can be done in a preprocessing step that costs $O(UM^2)$: if we fix i and any two breakpoints values, b_i, b_{i-1}, then all in-between ϕ's are equal and determined by i. As a result, ϕ_m comes out of the partial sum, and their contribution to the cost $(g(b_i, b_{i-1}))$ can be determined from the partial sums of the queue lengths; the latter can all be predetermined in $O(M^2)$. Therefore, all $g(b_i, b_{i-1})$'s (for all i's and all b_i, b_{i-1}'s) can be determined in $O(UM^2)$, and the overall complexity is $O(UM^2)$.

We underline that our algorithm takes into account the bandwidth allocation at the physical layer and produces a code allocation which can be implemented exactly. In fact, our solution capitalizes on the finite number of available codes in order to decrease complexity. Furthermore, our approach provides both "soft" and "hard" guarantees. "Hard" guarantees correspond to the lower bound throughput (which can be translated to delay guarantees if there is information on how traffic is generated by the sessions). "Soft" guarantees are provided in the sense that we go after minimizing a metric representative of the average evacuation delay in the system. Moreover, it should be mentioned that our scheme does not require statistical/deterministic description of the incoming traffic (this could be attractive especially in emerging third-generation networks where it is difficult to predict what will be the data requirements of mobile users).

D. Related Work

With respect to related work, there is indeed an extensive body of work on QoS scheduling in wireline networks (see, e.g., [23] and references therein), and especially FQ algorithms. Closed-form real-valued solutions for (2), (3) in the special case where $L_m = 1, U_m = C, \forall m$ have been derived in [24]. However, real ϕ_m's do not translate to multicode CDMA networks without approximation errors. Furthermore, [24] does not provide any hard QoS guarantees; hence, without a traffic policing mechanism, greedy sessions could monopolize all the bandwidth, and constant-bit-rate (CBR) sessions could be starved. Having in mind than in our scheduling framework, with proper setting of the weights and the constants $\{L_m, U_m\}_{m=1}^{M}$ *only one* session may transmit at a time, it is important to note that it has been advocated that only one user should be allowed to transmit at a time [4], [6]. In fact, [6], [7] and references therein present different criteria to select the *single* user i who is allowed to transmit: all these methods are proved to stabilize the system whenever a stable policy exists and, hence, these resource allocation methods are *throughput-optimal*. Our approach is different, as we allow many users to transmit simultaneously. Noting that it is possible to provide minimum rate guarantees, our scheduling algorithm can also result in queueing delays with smaller variance, compared to algorithms that allow only one user to transmit at a time. Essentially, the smaller variance of packet delays is a manifestation of the tradeoff between statistical multiplexing and partially fixed resource allocation methods (note that delay variance can be important in multimedia applications, because the delay variance—*jitter* is related to buffer requirements and selection of playback points). More details can be found in Section IV.

It is of interest to note that [25] is indeed formulated as a simultaneous multiple flow scheduling problem, but, as it turns out, throughput optimality can be attained by serving a single (properly selected) user per slot. Therefore, simultaneous multiflow scheduling is not necessary from a throughput viewpoint; yet, as we show in this paper, it is important for delay considerations. Also, the emphasis of [23], [25] is on stabilizability, while ours is on delay. We do not have any claims on maximizing stable throughput, but this is not the only consideration. Furthermore, in [23] and [25], a finite-state Markov channel is assumed, and certain technical conditions are placed on the arrival processes. We need neither of these. In short, the goal and flavor of our work and that of [23] and [25] are very different. Furthermore, in [23] the resulting optimal policies are more complex to implement than ours.

E. Latency Versus Average Throughput Tradeoff

As we saw in the previous section, the bounds $\{L_m, U_m\}_{m=1}^{M}$ serve two purposes: 1) they yield minimum throughput guarantees and 2) they guard the bandwidth allocation mechanism against malicious (greedy) data sources. It is interesting that these constants prove to be useful even under different operating assumptions. Up to this point, we have assumed a frequent code-reassignment procedure (which is based on a demand-assignment MAC protocol). It is well-known that demand-assignment MAC protocols offer bandwidth allocation flexibility at the expense of increased MAC overhead (mainly during the reservation phase). Suppose now that MAC overhead considerations do not allow frequent code reassignments, resulting in relatively long time intervals before new code allocations take effect. In such a scenario, the constants $\{L_m, U_m\}_{m=1}^{M}$ help to address the *latency versus average throughput* tradeoff, which was discussed in [26] for the high-data-rate CDMA/HDR system under a seemingly different context. Next, we revisit some of the results of [26], and see how they apply to our setup.

With $R_m = \alpha\phi_m$ (α a positive constant) denoting the transmission rate of user m, and P_m the proportion of the mth-user packets, the average throughput becomes

$$R_{\text{av}} = \sum_{m=1}^{M} P_m R_m.$$

As in [26], for user m the latency of the data packets is inversely proportional to R_m. On the other hand, if the same latency is to be provided to all users, then the average throughput is given from [26, Eq. (2)]

$$R'_{\text{av}} = \frac{1}{\sum\limits_{m=1}^{M} \frac{P_m}{R_m}}.$$

Playing with the parameters P_m and R_m, it is possible to come up with cases where R'_{av} is significantly smaller than R_{av}. For example, [26] reports that in the extreme case of a system with two users with $P_1 = P_2 = 0.5$ and $R_2 = 64R_1$, we can calculate $R'_{\text{av}}/R_{\text{av}} = 6\%$. Hence, the tradeoff between aggregate average throughput and latency. Note that the average throughput is a system-level parameter, whereas latency is related to the QoS that each user experiences.

The bounds $\{L_m, U_m\}_{m=1}^{M}$ help to address the aforementioned tradeoff. In the extreme case of $L_m = 1$, $U_m = C$, the scheduler will come up with a bandwidth allocation scheme where low-rate users experience high latencies. On the other hand, at the other extreme case where $L_m = U_m = C/M$, bandwidth allocation degenerates to all users having equal rates, and thus having equal latencies. Proper selection of the bounds $\{L_m, U_m\}_{m=1}^{M}$ appears to be an interesting problem. The selection depends, among other factors, on the pricing policy employed in the network (i.e., how revenue depends on satisfying individual/QoS requirements or the total amount of data traffic in the network).

F. Simulation Results

Let us present two examples of how our algorithm works when all users experience good channels. The first example shows how queueing delays can be significantly reduced. The second example shows that the code allocation process implicitly provides a mechanism to track the data transmission rates of mobile users.

Reducing Queueing Delays: We simulate a pico-cell where three mobile users communicate with the base station. We assume that $C = 32$ and that the traffic generated by each of the mobile users is Poisson with corresponding normalized rates $\lambda_1 = 1/2 - 1/128$, $\lambda_2 = 3/8 - 1/128$, and $\lambda_3 = 1/8 - 1/128$. In our first experiment, the initial weight assignment is $\beta_1 = 0.5$, $\beta_2 = 0.375$, and $\beta_3 = 0.125$ (under which, if all three users have data to transmit, users 1, 2, and 3 are assigned 16, 12, and 4 codes, respectively). The arrival rates are expressed in frames per transmission round, and we assume that $C = 32$ frames per transmission round can be transmitted (in Section IV-C we provide explicit numerical values). We simulate the system for 1000 transmission rounds, and Figs. 2–4 depict the number of queued packets and the number of allocated codes per user for a range of the transmission rounds. Fig. 2 corresponds to fixed ϕ's, whereas Figs. 3 and 4 illustrate how the queues sizes drop with our time-varying weights algorithm, when the DP algorithm for scheduling is run at each transmission round. In particular, Fig. 3 depicts the scenario under which $w_m = 1$, $\forall m$, whereas Fig. 4 depicts the case where $w_m = Q_m^{-1/2}$, $\forall m$ (when $Q_m \neq 0$). Comparing Fig. 3 to Fig. 4, it can be seen how peaks in the queue lengths are significantly reduced—this is because long queues are weighted more. Table I depicts the mean delays per packet (in transmission rounds) under the three schemes, where it is easily seen how the total average mean delay is reduced by our time-varying weights algorithm.

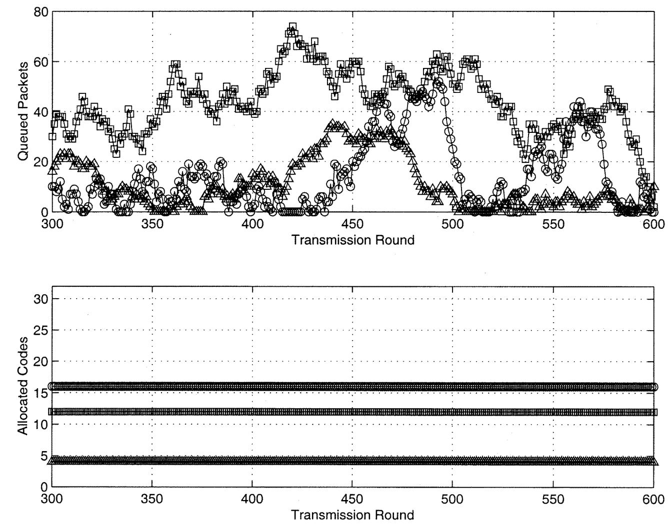

Fig. 2. Fixed ϕ's.

We have also considered a system with $C = 40$ codes and 16 users: eight high-rate users transmit Poisson traffic with $\lambda_{\text{hr}} = C/10 - C/128$, and eight low-rate users transmit Poisson traffic with $\lambda_{\text{lr}} = C/40 - C/128$. We simulate this system under FQ with fixed code allocation (whenever their respective queues are not empty, the high-rate users receive at least four codes, whereas the low-rate users receive at least one code). With fixed code assignment, the average queueing delays for the high-rate and the low-rate users are 0.934 and 1.09 slots, respectively. As in the previous example, our code allocation algorithm results in smaller packet delays: with time-varying weight assignment, the mean delays drop respectively to 0.343 and 0.006 slots when $w_m = 1$, and to 0.064 and 0.002 slots when $w_m = Q_m^{-1/2}$ (the codes are reallocated every transmission round). Note that with 16 users, queue length and code allocation plots become difficult to read. For this reason, we only report mean delay values.

Tracking of Input Rates: The time-varying weight assignment algorithm implicitly provides us with the opportunity of tracking the transmission rate of the sources. Fig. 5 illustrates such an example. We assume that mobile user m generates traffic which is the sum of a CBR source with rate $\lambda_m/2$ and a Poisson source with rate $\lambda_m/2$ (the λ's are set as in the

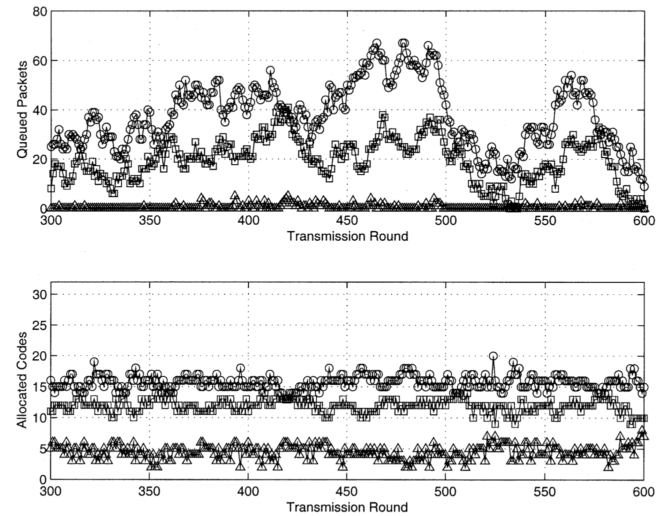

Fig. 3. Time varying ϕ's.

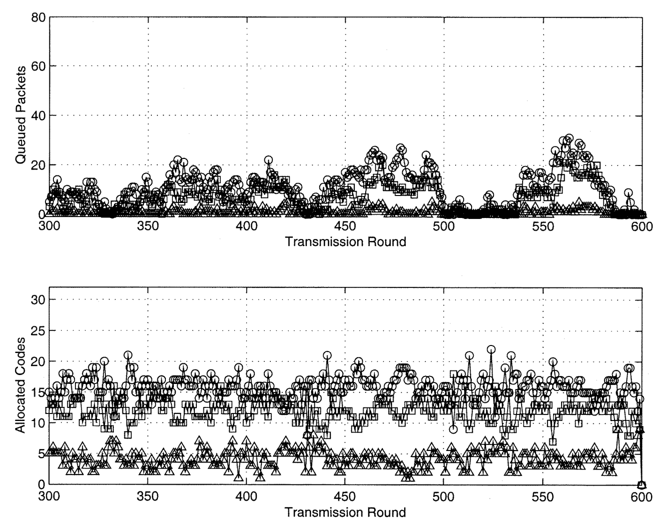

Fig. 4. Time varying ϕ's, weighted queues.

previous example). In other words, if we denote by $A_m(\tau, t)$ the amount of traffic that user m generates, then

$$A_m(\tau, t) = \frac{\lambda_m}{2}(t - \tau) + \text{Poisson}\left(\frac{\lambda_m}{2}; \tau, t\right).$$

We use the λ's of the previous example, but we assume that initially the base station does not know the λ's. The base station starts with the arbitrary code assignment of 10, 10, and 12 codes to the three users, respectively, and updates the ϕ's every ten transmission rounds. It can be seen from Fig. 5 than eventually users 1, 2, and 3 are assigned, respectively, (on the average) 16, 12, and 4 codes. This code assignment coincides with the long-term average transmission rate of the mobile users. Therefore, it can be seen that our time-varying weight assignment procedure can adapt to incoming traffic load.

IV. Ride the Wave

So far, we have seen how the bandwidth allocation procedure adapts to incoming traffic load. In this section, we discuss

TABLE I
Mean Delay Per Packet (in Transmission Rounds)

	Fixed ϕ's	Time-Varying $\phi's$	Weighted Queues ϕ's
user 1	0.8761	2.2305	0.6325
user 2	3.7277	1.5047	0.5089
user 3	2.1154	0.1052	0.1396

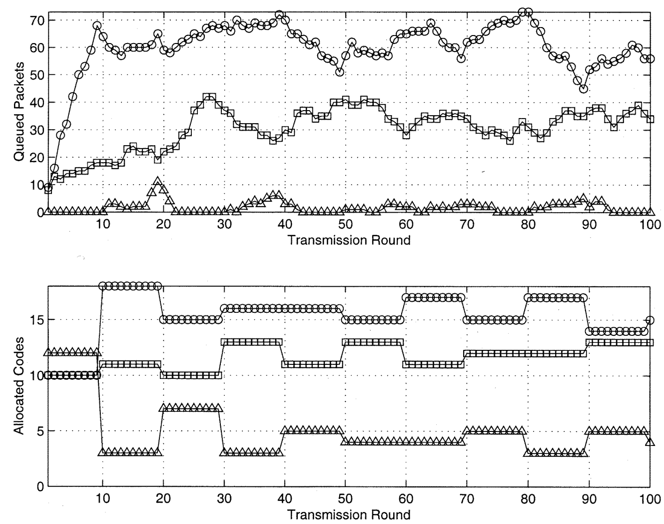

Fig. 5. Tracking input rates.

how the bandwidth allocation process may adapt to the underlying channel quality. The mechanism is rather intuitive: from a cell-wide throughput viewpoint, more codes should be assigned to users with "good" channels, and the bandwidth scheduler should try to "ride the wave." As the channel quality periodically varies, (ideally) at every time instant the system should allocate more bandwidth to users with good channels. Before we present the modified DP cost function and pertinent simulation results, we provide a brief overview of the *multiuser diversity*, which is inherent in a cellular environment (see [4] and references therein).

A. Multiuser Diversity

Throughout this section, we focus on the case where the power control is capable of compensating only the slow-fading component s_t, but it is not fast enough to track the fast-fading component h_t. Though at first sight, the presence of h_t appears to be an impediment to network performance, works such as [4] (and references therein) advocate that a system architect may build upon the SNR discrepancy and actually improve network performance. Herein, we are interested in the fact that as the number of users in the system increases, so does the probability that some of them will experience good channels. For convenience, we drop the time index t from h_t, and we look at a single cell with M users. We denote by $|h^{(m)}|$ the fading of the mth user. For Rayleigh fading, the complementary cumulative distribution function (CCDF) has the familiar form $F^{(m)}(h) \triangleq \text{Prob}(|h^{(m)}| > h) = e^{-h^2/2\sigma^2}$, $h \geq 0$. It is straightforward to see that the probability $\Phi^{(M,2)}(h)$

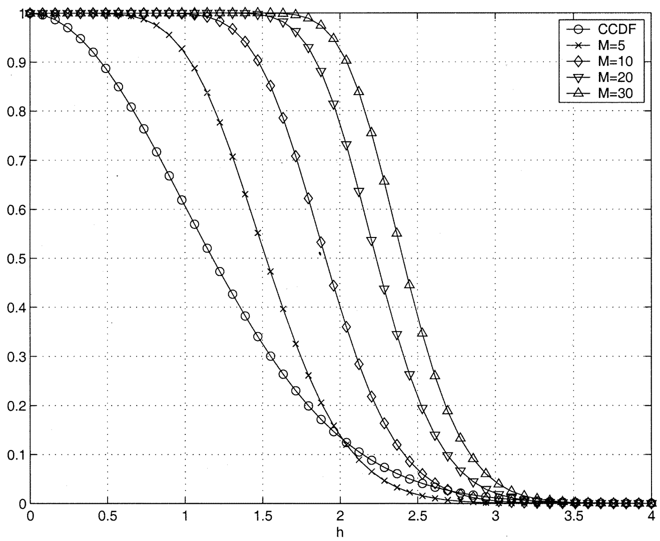

Fig. 6. Multiuser diversity: as the number of users M increases, so does the probability that at least some of them will have good channels.

that for at least two users, their respective fast-fading components are higher than the level h is given by $\Phi^{(M,2)}(h) = 1 - MF^{(m)}(h)(1-F^{(m)}(h))^{M-1} - (1-F^{(m)}(h))^M$. We plot in Fig. 6 the CCDF for one user, and the value of $\Phi^{(M,2)}$ for various values of M (with $\sigma = 1$). It is seen that as the number M of users increases, so does the probability that at least two of them experience a channel with fast-fading component higher than a prescribed threshold. Note that the result is trivially extended to an arbitrary number $K \leq M$, i.e., as M increases, the probability that at least K users experience good channels increases as well (if K is kept fixed).

However, a memoryless bandwidth scheduler (which makes decisions only by considering the instantaneous fading) could penalize stationary users who have poor channel quality. Hence, a *fair* bandwidth allocation process should not be memoryless; instead, it should account for the average number of codes that had been allocated to users in the recent past (see also the proportional fair scheduling approach of [4]). Summarizing, a time-varying wireless environment presents the opportunity to improve network performance by "riding the wave" at the potential expense of unfairness in the bandwidth allocation process. To address these issues, we modify the weights w_m in the DP cost function in Section IV-B. Before we present them, the following technical remark needs to be raised.

Throughout this section, we make the assumption that power control cannot always be perfect, thereby introducing a discrepancy of the channel quality for the various users in the network. This is not to say that our previous assumption on perfect power control is not reasonable. With fixed modulation, the combination of channel coding (which comes at the expense of data transmission rate) and spreading can mitigate the deleterious effects of the fast-fading component. With sufficient interleaving and powerful channel coding the transmission framework becomes very robust, even in harsh-fading conditions. However, the recent trend in wireless architectures is to improve performance by capitalizing on accurate SNR estimation, adaptive modulation, and adaptive channel coding (e.g., consider the various transmission modes in EDGE or the IEEE 802.11a standard). Although this operational scenario is more sensitive to estimation errors, it can potentially lead to improved performance. This well-known fact is further corroborated by our simulation results in Section IV-C.

B. Modified Weights

Recall from Section II-C that the cost function to be minimized is $\sum_{m=1}^{M}(Q_m/w_m\phi_m)$. We set the weights w_m as the product of three factors

$$w_m = \pi_m \bar{R}_m(t) \frac{1}{\log_2\left(1 + \frac{\mathrm{SNR}_m(t)}{\Gamma}\right)} \tag{4}$$

where

1) π_m is the priority assigned to user m;
2) $\bar{R}_m(t)$ is the average amount of data traffic of user m up to time t;
3) $\log_2(1+(\mathrm{SNR}_m(t)/\Gamma))$ denotes the expected normalized throughput for each code assigned to user m. With SG the spreading gain, $\mathrm{SNR}_m(t) = (\mathrm{SG}|h_m(t)|^2/\eta)$ is the signal-to-noise ratio for user m (in the absence of interferers). It follows from the Shannon capacity formula that $\log_2(1+(\mathrm{SNR}_m(t)/\Gamma))$ represents a measure of achievable normalized throughput per CDMA spreading code. In practice, we need to take into account that the number of information symbols that can be received correctly is less than the theoretical capacity—hence the factor Γ, which is known as the SNR gap [28].

Note that with $\pi_m = 1$, the weights take the form $\mathrm{achievable\ rate/average\ rate}$, which is reminiscent of the metric used by the scheduler in [4] and [29] that imposes a TDMA-like access mechanism, where only one user is allowed to transmit at a time. Herein lies an important difference with our approach. Our framework is more general, and allows the simultaneous transmission by many users. Indeed, from an information theory point of view, the TDMA-like mechanism maximizes the long-term average throughput [4], but it ignores short-term throughput and latency requirements. This is because [4] and [29] allow *only one* user to transmit, thereby shutting off users with relatively "bad" channels. However, short-term throughput and latency requirements are important for time-critical data applications (such as multimedia streaming). We underline that these requirements can be provided by our framework with proper selection of the priority weights π_m and the constants $\{L_m, U_m\}_{m=1}^{M}$. Additionally, in cases where the scheduler cannot reassign bandwidth very frequently (for reasons discussed in Section III-E), the capability of simultaneous transmissions becomes beneficial.

C. Simulation Results

To illustrate how network performance is improved by incorporating channel information into our DP-based scheduling framework, we need to describe our assumptions about the underlying CDMA physical layer, and fill in some important details about the radio interface. We consider a CDMA

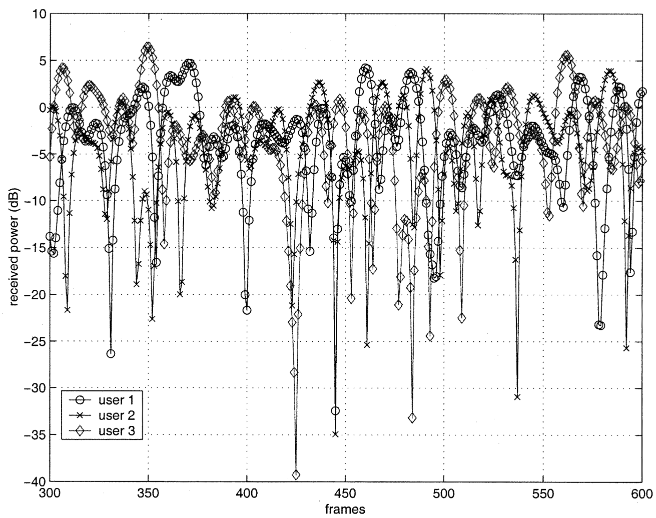

Fig. 7. Fast-fading component of mobile users during frame transmission.

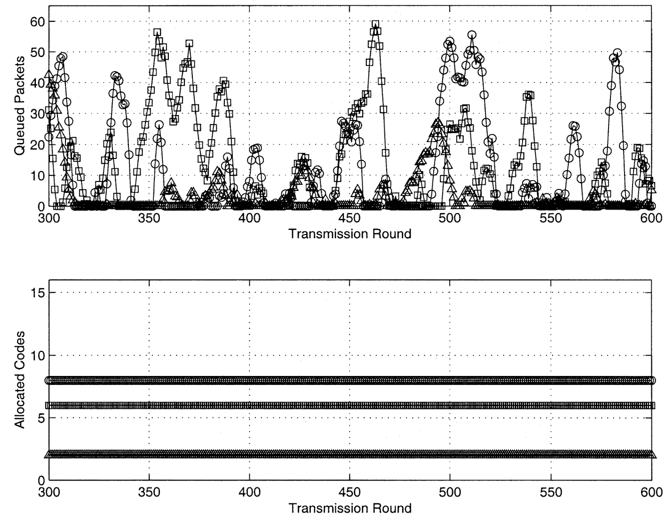

Fig. 8. Three users with fixed ϕ's.

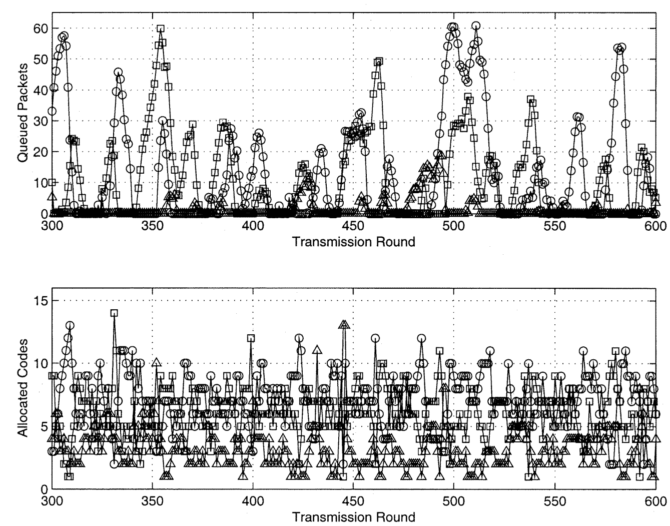

Fig. 9. Three users with time varying ϕ's.

system that operates at the carrier frequency of 1 GHz with spreading bandwidth 1.2288 MHz. The spreading factor is 16; hence, 76 800 symbols/s can be transmitted using each of the 16 available CDMA codes. With normalized noise power $\eta = 1$ (which follows from our operating assumptions on power control), the normalized throughput per CDMA code is $C_{\rm norm} = \boldsymbol{E}\{\log_2(1 + (16\|h\|^2/\Gamma))\}$ with the expectation taken over h. With $\Gamma = 6$ dB, and h having Rayleigh envelope, the normalized throughput is numerically evaluated to be $C_{\rm norm} = 1.9395$ information bits per transmission. If we treat each CDMA code as a separate channel, then our CDMA system appears to have the capability of transmitting $16 \cdot 76\,800 \cdot 1.9395 = 2.38$ Mb/s of data traffic.

Similar to the first example of Section III-F, we simulate a system with three users. The users generate Poisson data traffic with average rates $\lambda_1 = 280$, $\lambda_2 = 210$, and $\lambda_3 = 70$ packets per second, respectively. Each packet is 512 bytes, which makes the total average traffic load $(280 + 210 + 70) \cdot 8 \cdot 512 = 2.29$ Mb/s. Suppose that the frame length is 512 (coded) symbols, which are transmitted in $(512/76\,800) = 6.67$ ms. If the mobile users are slowly moving at the speed of 5 mph, then the Doppler is evaluated to be 7.45 Hz and the channel coherence time 0.134 s. Hence, it is expected that after the transmission of $(0.134\ \text{s}/6.67\ \text{ms}) \approx 20$ frames, the channels will have changed significantly. Fig. 7 depicts sample paths of the stochastic processes $|h_1(t)|^2$, $|h_2(t)|^2$, and $|h_3(t)|^2$ that correspond to the three users in our CDMA pico-cell (the plot was generated by applying the Jakes model). The fluctuations of the received signal power are clearly illustrated, and so are the opportunities for "riding the wave." Since the underlying channels are independent, the power valleys do not coincide in time, thus making it possible to allocate more codes to the user with the strongest channel. Additionally, higher SNR values allow faster transmission rates.

We feed the sample paths of Fig. 7 into our scheduling algorithm, which makes code assignment decisions on a per frame basis using the weights of (4). We assume perfect SNR estimation, and that there is fine granularity of available transmission modes, which allows each frame to carry $512 \cdot \log_2(1 + (16\|h\|^2/\Gamma))$ information symbols. Figs. 8 and 9 depict the number of queued packets and the number of allocated codes per user for a range of transmission rounds. Fig. 8 corresponds to the case where ϕ's are fixed, but the transmissions rates (per CDMA code) vary depending on the perfectly known instantaneous SNR. On the other hand, Fig. 9 corresponds to the case where transmission rates also vary as a function of the SNR, and the ϕ's are given by the DP algorithm (which takes into account the received powers). The effectiveness of the DP algorithm can be seen from Table II,

TABLE II
MEAN DELAY PER PACKET (IN TRANSMISSION ROUNDS)

	Fixed ϕ's	Time-Varying ϕ's	Weighted Queues ϕ's	HDR-like	EXP
user 1	1.0116	1.2616	1.1311	42.3951	2.2792
user 2	2.0204	1.6627	1.6355	29.9905	3.2497
user 3	7.1404	0.9789	0.3063	2.6643	8.2702

which contains average values for the mean delay for a total of 1000 transmission rounds, and shows that the combination of various transmission modes[6] and judicious code allocation boosts network performance by decreasing significantly the queueing delays. Furthermore, the effect of incorporating the underlying channel quality in the DP algorithm becomes obvious if we compare the allocated number of codes in Figs. 3 and Fig. 9. In Fig. 9 the number of codes allocated to each user varies significantly more than that depicted in Fig. 3. This wide variation is a direct result of the definition of weights in (4) and the fast-varying channel quality—with the modified weights, the DP algorithm forces the bandwidth allocation procedure to track the received power fluctuations and thereby to "ride the wave."

Next, we compare our framework to two schedulers that at time slot t allocate all CDMA codes to a single user m. The first scheduler uses the criterion

$$\text{choose user } m \text{ with the highest } \frac{\log_2\left(1+\frac{\mathrm{SNR}_m(t)}{\Gamma}\right)}{\bar{R}_m(t)} \tag{5}$$

with $\bar{R}_m(t)$ the average rate allocated to user m up to time t. The second scheduler uses the criterion

$$\text{choose user } m \text{ with the highest}$$
$$\log_2\left(1+\frac{\mathrm{SNR}_m(t)}{\Gamma}\right)\exp\left(\frac{Q_m}{1+\sqrt{\frac{1}{M}\sum_{\mu=1}^{M} Q_\mu}}\right). \tag{6}$$

The first allocation policy follows the doctrine of [4] and [29], whereas the second scheduler is discussed in [6] (and references therein).[7] It is important to recall that these schedulers were designed under different design criteria. Ours minimizes the average evacuation delay while providing minimum throughput guarantees, whereas the scheduler as in [4] and [29] maximizes the long-term aggregate network data throughput. On the other hand, the scheduler in [6] is also proved to be throughput optimal but, compared to the HDR scheduler, also reduces queueing delays.

From a certain viewpoint, it is not fair to compare these schedulers, because they have been designed with an eye to different criteria. However, since in this work we are primarily interested in minimizing delay, it is of high interest to present the delay-performance of a throughput-maximizing scheduler. This is depicted in Figs. 10 and 11 and quantified in the two right-most columns of Table II. Similar to Figs. 8 and 9, Figs. 10 and 11 depict the queue lengths per user (note that the number of allocated codes can be either 16 or zero) under the scheme of [4], [29], and [6], respectively. If we compare Fig. 9 to Fig. 10, we can see how the queue lengths have increased by almost an order of magnitude. Also, if we take a closer look at Table II, we can see how that average delays have been increased significantly under the scheduling policy like in [4] and [29]. We underline that this is *not* an unexpected result, because the scheduling policy like in [4] and [29] ignores data packet delays. On the other hand, if we compare Fig. 11 to Fig. 10, we observe that indeed the criterion (6) significantly reduces the queue lengths and also reduces the average mean delay (as evinced by the last column of Table II). However, it is seen that criterion (6) results in *longer* queues, and *higher* mean delay than that of our scheme. Finally, Table III depicts the packet delay variance (jitter) of the aforementioned algorithms, where it is seen that our resource allocation method results in smaller values.

Overall, this simulation example illustrates the merits of our approach in cases where delay guarantees and smaller delay variance are important (such as in streaming applications that deliver multimedia content to mobile users).

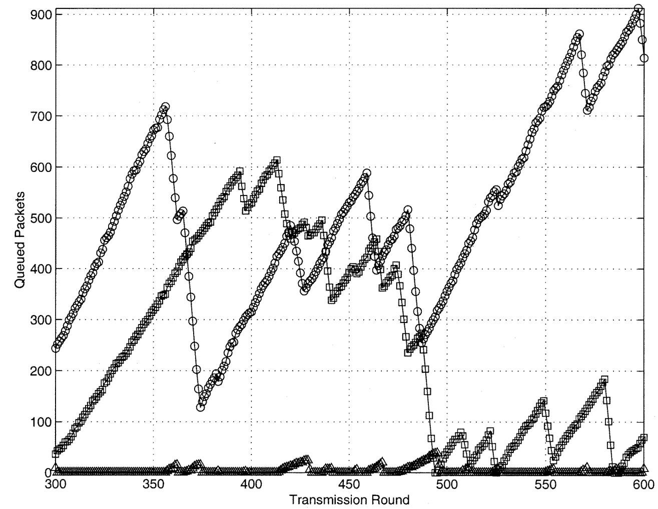

Fig. 10. HDR-like scheduler.

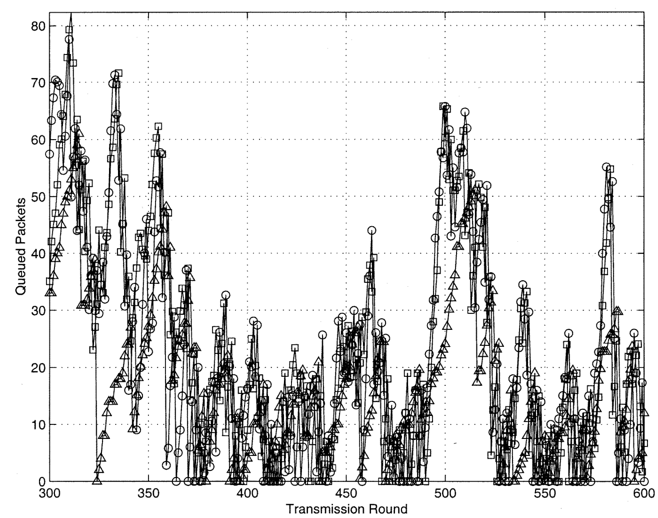

Fig. 11. Throughput-optimal exponential rule.

[6]However, it is important to note that the aforementioned assumptions on achievable data rate are optimistic since they depend on perfect SNR estimation, and ignore rate quantization (because in real systems the offered transmission rates comprise a rather small discrete set, depending on modulation formats and channel coding rates). We have made these operating assumptions in order to bring to surface the potential gains that our DP-based framework yields. Certainly, SNR estimation errors and the discrete nature of available transmission modes will have a negative effect on the improvements of a time-varying bandwidth allocation policy. Quantifying these negatives effects is a subject that warrants thorough investigation, which is beyond the scope of this paper.

[7]Note that we have set the constants $\gamma_m = \alpha_m = 1$. These constants are defined in [6] to weigh differently the various achievable rates, and the queueing delays. We have chosen to set $\gamma_m = 1$ in order to treat achievable rates in a uniform way, and $\alpha_m = 1$, following the rule of thumb $\alpha_m = -\log(\delta_m)/T_m$, where δ_m is the probability that user m will experience delay larger than T_m slots. For further details, see [6] and references therein.

TABLE III
DELAY VARIANCE PER PACKET (IN TRANSMISSION ROUNDS)

	Fixed ϕ's	Time-Varying $\phi' s$	Weighted Queues ϕ's	HDR-like	EXP
user 1	3.0138	3.7487	3.3339	1116.7	5.5851
user 2	9.5832	7.9753	7.9005	789.9	11.2428
user 3	115.28	4.0529	5.6939	19.8	51.8296

V. CONCLUSION

We have presented a dynamic programming algorithm which solves the problem of bandwidth allocation in fair queueing wireless networks with an underlying multicode CDMA physical layer. Our approach is based on a time-varying weight assignment, which minimizes the queueing delays of mobile users, while providing both "soft" and "hard" QoS guarantees. Our computationally efficient algorithm produces the exact discrete solution, obeys constraints imposed by the underlying physical layer, and, in fact, capitalizes on the discrete nature of the available service rates to reduce complexity. Furthermore, in cases where information about the underlying channel quality is available, our algorithm exploits the inherent multiuser diversity by "riding the wave": the DP-based bandwidth allocation mechanism assigns more CDMA codes to users with "good" channels without, however, shutting off users with relatively "bad" channels. In both cases, we presented simulation results that illustrated the potential of our approach. Future research includes an analytical study of the properties of our time-varying scheduler, and extension of the code allocation mechanism to multicell environments, where interference from neighboring cells needs to be taken into account.

REFERENCES

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

[2] A. Stamoulis and G. B. Giannakis, "Packet fair queueing scheduling based on multirate multipath-transparent CDMA for wireless networks," in *Proc. IEEE INFOCOM*, Tel Aviv, Israel, Mar. 2000, pp. 1067–1076.

[3] Y. Cao and V. O. K. Li, "Scheduling algorithms in broadband wireless networks," *Proc. IEEE*, vol. 89, pp. 76–87, Jan. 2001.

[4] P. Viswanath, D. Tse, and R. Laroia, "Opportunistic beamforming using dumb antennas," *IEEE Trans. Inform. Theory*, vol. 48, pp. 1277–1294, June 2002.

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

[6] S. Shakkottai, R. Srikant, and A. Stolyar, "Pathwise optimality and state space collapse for the exponential rule," in *Proc. ISIT*, Lausanne, Switzerland, June/July 2002, p. 379.

[7] M. Andrews, K. Kumaran, K. Ramanan, A. Stolyar, P. Whiting, and R. Vijayakumar, "Providing quality of service over a shared wireless link," *IEEE Commun. Mag.*, vol. 39, pp. 150–154, Feb. 2001.

[8] A. Demers, S. Keshav, and S. Shenker, "Analysis and simulation of a fair queueing algorithm," in *Proc. ACM SIGCOMM*, 1989, pp. 1–12.

[9] D. E. Wrege, "Multimedia networks with deterministic quality-of-service guarantees," Ph.D. dissertation, Univ. of Virginia, Charlottesville, VA, Aug. 1996.

[10] R. L. Cruz, "A calculus for network delay, part I: network elements in isolation," *IEEE Trans. Inform. Theory*, vol. 37, pp. 114–131, Jan. 1991.

[11] Y. Zhou and H. Sethu, "On the relationship between absolute and relative fairness bounds," *IEEE Commun. Lett.*, vol. 6, pp. 37–39, Jan. 2002.

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

[13] R. Szabo, P. Barta, J. Biro, F. Nemeth, and C.-G. Pentz, "Non-rate-proportional weighting of generalized processor sharing schedulers," in *Proc. IEEE GLOBECOM*, Rio de Janeiro, Brazil, Dec. 1999, pp. 1334–1339.

[14] A. Elwalid and D. Mitra, "Design of generalized processor sharing schedulers which statistically multiplex heterogeneous QoS classes," in *Proc. IEEE INFOCOM*, 1999, pp. 1220–1230.

[15] S. Zhou, G. B. Giannakis, and C. Le Martret, "Chip-interleaved block-spread code division multiple access," *IEEE Trans. Commun.*, vol. 50, pp. 235–248, Feb. 2002.

[16] Z. Wang and G. B. Giannakis, "Wireless multicarrier communications: Where Fourier meets Shannon," *IEEE Signal Processing Mag.*, vol. 17, pp. 29–48, May 2000.

[17] G. J. Foschini and Z. Miljanic, "A simple distributed autonomous power control algorithm and its convergence," *IEEE Trans. Veh. Technol.*, vol. 42, pp. 641–646, Nov. 1993.

[18] W. C. Y. Lee and Y. S. Yeh, "On the estimation of the second-order statistics of lognormal fading in mobile radio environment," *IEEE Trans. Commun.*, vol. COM-22, pp. 869–873, June 1974.

[19] J. C. Chen, K. M. Sivalingam, and R. Acharya, "Comparative analysis of wireless atm channel access protocols supporting multimedia traffic," *Mobile Networks and Applications*, vol. 3, no. 3, pp. 293–306, 1998.

[20] L. Tassiulas, "Cut-through switching, pipelining, and scheduling for network evacuation," *IEEE/ACM Trans. Networking*, vol. 7, pp. 88–97, Feb. 1999.

[21] R. Bellman, *Applied Dynamic Programming*. Princeton, NJ: Princeton Univ., 1962.

[22] N. D. Sidiropoulos and R. Bro, "Mathematical programming algorithms for regression-based nonlinear filtering in $\mathbb{R}^N$," *IEEE Trans. Signal Processing*, vol. 47, pp. 771–782, Mar. 1999.

[23] A. Eryilmaz, R. Srikant, and J. Perkins. (2002) Stable scheduling policies for fading wireless channels. [Online]. Available: http://tesla.csl.uiuc.edu/-srikant/pub.html

[24] H.-T. Ngin, C.-K. Tham, and W.-S. Sho, "Generalized minimum queueing delay: an adaptive multi-rate service discipline for ATM networks," in *Proc. IEEE INFOCOM*, 1999, pp. 398–404.

[25] S. Shakkottai and A. Stolyar. (2001) Scheduling for multiple flows sharing a time-varying channel: the exponential rule. *Translations of the AMS, a Volume in Memory of F. Karpelevich* [Online]. Available: http://www.ece.utexas.edu/~shakkot/Pubs/exfluid.ps

[26] P. Bender, P. Black, M. Grob, R. Padovani, N. Sindhushyana, and A. Viterbi, "CDMA/HDR: a bandwidth efficient high-speed wireless data service for nomadic users," *IEEE Commun. Mag.*, vol. 38, pp. 70–77, July 2000.

[27] D. Bertsekas and R. Gallager, *Data Networks*. Englewood Cliffs, NJ: Prentice-Hall, 1992.

[28] J. M. Cioffi, "A multicarrier primer," ANSI, TIEI.4 Contribution 91.157, [Online.] Available: http://isl.stanford.edu/~cioffi/papers.html, Nov. 1991.

[29] A. Jalali, R. Padovani, and R. Pankaj, "Data throughput of CDMA-HDR, a high efficiency-high data rate personal communication wireless system," in *Proc. Veh. Technol. Conf.*, Tokyo, Japan, 2000, pp. 1854–1858.

Anastasios Stamoulis received the Diploma degree in computer engineering from the University of Patras, Patras, Greece, in July 1995, the Master of Computer Science degree from the University of Virginia, Charlottesville, in May 1997, and the Ph.D. degree in electrical engineering from the University of Minnesota, Minneapolis, in December 2000.

He is now with Qualcomm, Inc., San Diego, CA.

Nicholas D. Sidiropoulos (M'92–SM'99) received the Diploma degree in electrical engineering from the Aristotelian University of Thessaloniki, Thessaloniki, Greece, and the M.S. and Ph.D. degrees in electrical engineering from the University of Maryland, College Park (UMCP), in 1988, 1990, and 1992, respectively.

From 1988 to 1992, he was a Fulbright Fellow and a Research Assistant at the Institute for Systems Research (ISR), UMCP. From September 1992 to June 1994, he served as a Lecturer with the Hellenic Air Force Academy. From October 1993 to June 1994, he was a Member of the Technical Staff, Systems Integration Division, G-Systems Ltd., Athens, Greece. From 1994 to 1995, he held a Postdoctoral position and, from 1996 to 1997, a Research Scientist position at ISR-UMCP before joining the Department of Electrical Engineering, University of Virginia, Charlottesville, in July 1997 as an Assistant Professor. He is currently an Associate Professor with the Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis. His current research interests are primarily in multiway analysis and its applications in signal processing for communications networking.

Dr. Sidiropoulos is a member of the Signal Processing for Communications Technical Committee (SPCOM-TC) of the IEEE Signal Processing Society and currently serves as Associate Editor for IEEE TRANSACTIONS ON SIGNAL PROCESSING and the IEEE SIGNAL PROCESSING LETTERS. He received the National Science Foundation CAREER Award in the Signal Processing Systems Program in June 1998.

Georgios B. Giannakis (F'97) received the Diploma in electrical engineering from the National Technical University of Athens, Athens, Greece, in 1981. He received the M.Sc. degree in electrical engineering in 1983, the M.Sc. degree in mathematics in 1986, and the Ph.D. degree in electrical engineering in 1986, from the University of Southern California (USC), Los Angeles.

After lecturing for one year at USC, he joined the University of Virginia, Charlottesville, in 1987, where he became a Professor of electrical engineering in 1997. Since 1999, he has been a Professor with the Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis, where he now holds an ADC Chair in Wireless Telecommunications. His research interests span the areas of communications and signal processing, estimation and detection theory, time-series analysis, and system identification—subjects, on which he has published more than 150 journal papers, 300 conference papers, and two edited books. His current research topics focus on transmitter and receiver diversity techniques for single- and multi-user fading communication channels, complex-field and space-time coding for block transmissions, multi-carrier, and ultrawide band wireless communication systems. He is a frequent consultant for the telecommunications industry.

Dr. Giannakis is the corecipient of four best paper awards from the IEEE Signal Processing Society in 1992, 1998, 2000, and 2001. He also received the Society's Technical Achievement Award in 2000. He co-organized three IEEE-SP Workshops, and guest coedited four special issues. He has served as Editor-in-Chief for the IEEE SIGNAL PROCESSING LETTERS, as Associate Editor for the IEEE TRANSACTIONS ON SIGNAL PROCESSING, and the IEEE SIGNAL PROCESSING LETTERS, as Secretary of the Signal Processing Conference Board, as a member of the Signal Processing Publications Board, a member and Vice-Chair of the Statistical Signal and Array Processing Technical Committee, and as Chair of the Signal Processing for Communications Technical Committee. He is a member of the Editorial Board for the PROCEEDINGS OF THE IEEE and the steering committee of the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS. He is a member of the IEEE Fellows Election Committee and the IEEE Signal Processing Society's Board of Governors.