TIME-VARYING FAIR QUEUING SCHEDULING FOR MULTICODE CDMA BASED ON DYNAMIC PROGRAMMING

A. Stamoulis

AT&T Labs-Research
180 Park Av, Florham Park, NJ 07932
as@research.att.com

N. Sidiropoulos and G.B. Giannakis

Dept. of ECE, University of Minnesota
200 Union Street SE, Minneapolis, MN 55455
{nikos,georgios}@ece.umn.edu

Abstract

Fair Queuing (FQ) algorithms, which have been proposed for QoS wireline-wireless networking, rely on the fundamental idea that the service rate allocated to user m is proportional to a positive weight ϕ_m. Targeting wireless data networks with a multicode CDMA-based physical layer, we develop FQ with time-varying weight assignments in order to minimize the queuing delays of mobile users. Applying dynamic programming, we design a computationally efficient algorithm which produces the optimal service rates while obeying i) constraints imposed by the underlying physical layer, and ii) QoS requirements. Simulations illustrate the merits of our designs.

1. 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. This is because network scheduling algorithms play a central role in how bandwidth is allocated among sessions–flows (in a wireline environment) or users (in a cellular network). In wired networks, the scheduling algorithm determines the transmission order of packets in outgoing links and it has a direct impact on the packet delay and achievable throughput (which serve as primary figures of merit of the system performance); hence, the extensive body of work on scheduling for wireline Computer Networks. In wireless networks, the scheduler (which resides at the base-station in a centralized implementation) allocates bandwidth by, e.g., assigning slots (in TDMA environments) or codes (in 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 broadband 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 Queuing (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 broadband wireless networks (see, e.g., [2] and references therein). The fundamental notion in GPS-based algorithms is that the amount of service session i receives from the switch (in terms of transmitted packets) is proportional to a positive weight ϕ_i. 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 for leaky-bucket constrained sessions [1], or stochastic, as in, e.g., [3]).

One of the major shortcomings of GPS is that the service guarantees provided to a session i are controlled by just one parameter, the weight ϕ_i. 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 work, we look at the problem of minimizing queuing 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, we design a computationally efficient algorithm, which produces the optimal weights ϕ_m's, that minimize a cost function representing the queuing 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.

The rest of this paper is structured as follows: Section 2 states the problem and describes our modeling assumptions. Section 3 presents our algorithm, and proves its optimality, while Section 4 illustrates the merits of our approach through simulations. We conclude this paper and give pointers to future research in Section 5.

2. 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 queuing algo-

N. Sidiropoulos was supported by NSF/Wireless CCR-9979295.

0-7803-7206-9/01/$17.00 © 2001 IEEE

rithm, decides on the corresponding CDMA codes, and communicates bandwidth/code assignments to mobile users using a demand assignment 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.

2.1. Fair Queuing Scheduler

The bandwidth allocation policy is based on the Generalized Processor Sharing (GPS) scheduling algorithm [1], which is well known for its perfect isolation and perfect fairness properties. GPS, also known as Weighted Fair Queuing [4], possesses properties which are attractive from both a network-wide point of view, and from a user perspective. 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: i) network resources are allocated irrespective of the behavior of the other sessions (which refers to the *isolation* property of the scheduler), and ii) 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. Each session i is characterized by a positive constant ϕ_i, and the amount of service $R_i(\tau,t)$ session i receives in the interval $(\tau,t]$ is proportional to ϕ_i, provided that the session is continuously backlogged. Formally, under GPS, if session i is continuously backlogged in $(\tau,t]$, then it holds:

$$\frac{R_i(\tau,t)}{R_j(\tau,t)} \geq \frac{\phi_i}{\phi_j} ,$$

for all sessions j that have also received some service in this time interval. It follows that in the worst case, the minimum guaranteed rate g_i given to session i is $g_i = r\phi_i / \sum_{j=0}^{N-1} \phi_j$, where N is the maximum number of sessions that could be active in the system. Therefore, a lower bound for the amount of service that session i is guaranteed is: $R_i^l(\tau,t) := (t-\tau) r\phi_i / \sum_{j=0}^{N-1} \phi_j$. If session i is (σ_i, ρ_i)-leaky bucket constrained[1], and the minimum guaranteed rate is such that $g_i \geq \rho_i$, then the maximum delay is $D_i^{\max} \leq \sigma_i / g_i$ (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 FDMA physical layer multiplexing techniques. What is radically different about GPS is its perfect fairness property: whenever a session i generates traffic at a rate less than g_i, 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 $\phi_{hr} = 1$, and the two (low-rate) users have a weight of $\phi_{lr} = 0.5$. When all users are active, the high-rate user will take 50% of the bandwidth, and each of 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 a multiple of 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 [6], 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" (SCR) in ATM terminology), then the allocation of ϕ'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., [7]), it is still considered quite challenging (see, e.g., [8]). One of the contributions of this work is that our algorithm yields the optimal set of weights which minimizes a cost function representing the queuing delays of mobile users. Before we give a mathematical description of this cost function in Section 2.3, let us cast an eye to how bandwidth is actually allocated at the physical layer.

2.2. Bandwidth Allocation under Multicode CDMA

At the physical layer, we assume a multicode CDMA transmission/reception scheme: there are C available codes[2] (these codes could be, e.g., Pseudo-Noise or Walsh-Hadamard), which can be allocated to mobile users. Each user m is allocated c_m codes, and splits the information stream into c_m substreams which are transmitted simultaneously using each of the c_m codes: it readily follows that if user m has Q_m data symbols to transmit, then Q_m/c_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. For example, the aforementioned assumption would entail (see, e.g, [9]):

$$\frac{(\frac{p_m}{c_m})\mathbf{s}_{m,c}^{\mathcal{H}}\mathbf{s}_{m,c}}{\sum_{\mu,c',(\mu,c')\neq(m,c)}^{M,c_\mu} (\frac{p_\mu}{c_\mu})\mathbf{s}_{m,c}^{\mathcal{H}}\mathbf{s}_{\mu,c'} + \eta} > \gamma , \quad \begin{matrix} 1 \leq m \leq M \\ 1 \leq c \leq c_m \end{matrix} , \qquad (1)$$

in the case where the conventional matched filter detector is employed at the basestation, user m uses codes with signatures $\{\mathbf{s}_{m,c}\}_{c=1}^{c_m}$, η is the variance of the additive noise, p_m,g_m are respectively the power, channel-gain for user m, and γ is the minimum required SNR for a prescribed bit error probability.

Unlike wired networks, the implementation of GPS in multicode CDMA networks appears to be straightforward[3]: given the assumption on the successful use of all C codes, c_m/C essentially denotes the bandwidth which is allocated to user m, and GPS is implemented by setting

$$c_m = \lfloor \frac{\phi_m}{\sum_{\mu\in\mathcal{A}} \phi_\mu} C \rfloor , \qquad (2)$$

where $\mathcal{A}$ is the set of active users (note that with C sufficiently large and frequent code re-assignments, the approximation error in implementing GPS using (2) can be made very small).

[1] i.e., for every interval $(\tau,t]$, the amount of traffic that session i generates is upper bounded by $\sigma_i + \rho_i(t-\tau)$ [5].

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

[3] In wired networks, real-world routers operate at the packet or cell level, whereas GPS assumes a fluid model of traffic. Hence, in practice GPS needs to be approximated by a Packet Fair Queuing (PFQ) algorithm [1]. Starting with Weighted Fair Queuing (WFQ) [4], there has been a lot of work on approximating GPS (see, e.g., [10–14]).

Finally, let us comment on the fact that 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 explain later on); the base station calculates the c_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) multimedia information. Note that i) the duration of the reservation phase can be reduced if users piggy-back their queue-lengths in prespecified intervals), and that ii) 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 the them (see, e.g., [15] for details).

2.3. Problem Statement

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

$$\text{minimize} \sum_{m=1}^{M} \frac{Q_m}{w_m \phi_m} \text{ subject to} \tag{3}$$

$$\sum_{m=1}^{M} \phi_m = C\ ,\ 1 \leq L_m \leq \phi_m \leq U_m \leq C\ ,\ \phi_m \in \mathbb{N}\ . \tag{4}$$

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 queuing delays is minimized. In particular, $\sum_{m=1}^{M} Q_m/\phi_m$ can be thought of as the total delay in the system[4], if no more packet arrivals occur. The constants $w_m > 0$, $1 \leq m \leq M$, allow us to introduce different priorities in the system: low (high) priority users should be assigned constants w_{lp} (w_{hp}) with $w_{lp} > w_{hp}$. 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 the bandwidth. We remark 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 (3).

3. DYNAMIC PROGRAMMING-BASED SOLUTION

In general, dynamic programming (DP) [16] can be used to search for the M-tuple $\{x_m\}_{m=1}^{M}$ of finite-alphabet "state" variables that minimizes $\sum_{m=1}^{M} \text{cost}_m(x_m, x_{m-1})$, where x_0 is given and

[4]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. As transmissions are allowed to overlap, the delay in the system is*not* $\sum_{m=1}^{M} Q_m/\phi_m$; rather, the delay is upper bounded by $\max_{1\leq m\leq M}\{Q_m/\phi_m\}$. In fact, the delay can be made much smaller than $\max_{1\leq m\leq M}\{Q_m/\phi_m\}$, because when the queue of the μ-th user drains out, the μ-th 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.

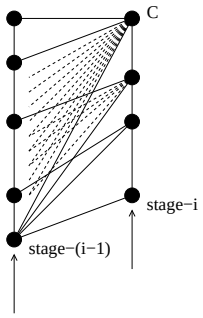

Fig. 1. Nodes (states) at stage i and predecessors at stage $(i-1)$

$\text{cost}_m(\cdot,\cdot)$ is some arbitrary "one-step transition" cost. 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 $\text{cost}_m(\cdot,\cdot)$ 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

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

We can enforce the sum constraint in (4) by demanding that the terminal state is $x_M = C$, and specifying $\text{cost}_m(x_m, x_{m-1}) = \infty$ whenever $x_m \leq x_{m-1}$. The inequality constraints in (4) can be enforced in a similar fashion by restricting the state "fan-out". Finally, for all allowable state transitions we set $\text{cost}_m(x_m, x_{m-1}) = \frac{Q_m}{w_m}\frac{1}{x_m - x_{m-1}}$.

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 $(i-1)$. At stage i, the node tags correspond to all possible values of $\sum_{m=1}^{i} \phi_m$, and similarly for stage $(i-1)$. At stage i, the bottom node has just one predecessor which is 1 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 2 nodes apart. If we take $\{L_m, U_m\}_{m=1}^{M}$ into account, then some transition steps are eliminated: in particular i)node n at stage $(i-1)$ is not allowed to lead to nodes $1, \ldots, n + L_i - 1$ of stage i, and ii)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 non-trivial QoS constraints, the execution time of the algorithm is decreased as transitions in the trellis diagram are expurgated.

However, when C is relatively large with respect to M, and $L_m = 1$, $U_m = U$, for all m, then there is an alternative $O(UM^2)$ solution. Following the material in [17], 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 big enough). Each breakpoint variable takes values in $1, \cdots, M$. 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 pre-determined 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 $O(UM^2)$.

Before we present simulation results, a few comments are in order. First, 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 total transmission delay in the system. Finally, we remark that our scheme does not require statistical/deterministic description of the incoming traffic (this could be attractive especially in emerging 3G networks where it is difficult to predict what will be the data requirements of mobile users).

With respect to related work, there is indeed an extensive body of work on QoS scheduling in wireline networks, and especially Fair Queuing algorithms. Especially for time-varying weight assignments, works such as [18] (and references therein) assume knowledge (in a deterministic or statistical sense) of incoming traffic. Work most similar in spirit to ours is that of [19], which has derived closed-form real-valued solutions for (3), (4) in the special case where $L_m = 1$, $U_m = C$, $\forall m$. However, real ϕ_m's do not translate to multicode CDMA networks without approximation errors. Furthermore, [19] does not provide any hard QoS guarantees: hence, without a traffic policing mechanism, greedy sessions could monopolize all the bandwidth, and constant-bit-rate sessions could be starved.

4. SIMULATIONS

We simulate a pico-cell where 3 mobile users communicate with the base-station. We assume $C = 32$, and that the traffic generated by each of the mobile users is Poisson with corresponding normalized rates $\lambda_1 = 1/2 - 1/128$, $\lambda_2 = 3/8 - 1/128$, $\lambda_3 = 1/8 - 1/128$. In our first experiment, the initial weight assignment is $\phi_1 = 0.5$, $\phi_2 = 0.375$, $\phi_3 = 0.125$ (under which, if all three users have data to transmit, users 1, 2, 3 are assigned 16, 12, and 4 codes respectively).

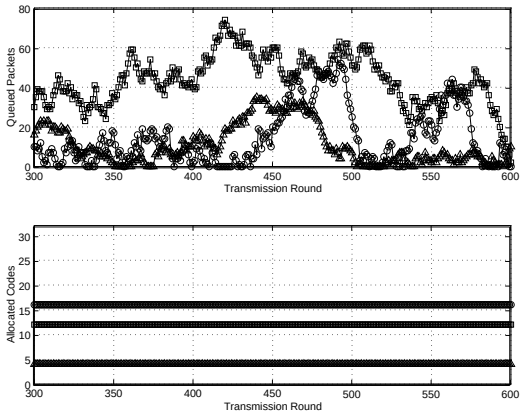

Fig. 2. Fixed ϕ's

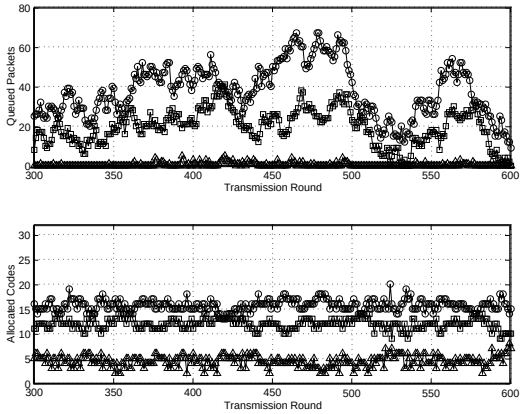

Fig. 3. Time varying ϕ's

We simulate the system for 1000 transmission rounds, and Fig. 2, 3, 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 Fig. 3, 4 illustrate how the queues sizes drop with our time-varying weights algorithm. In particular, Fig. 3 depicts the scenario under which $w_m = 1$, $\forall m$, whereas 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 (as long queues are weighted more). Table 1 depicts the mean delays per packet (in transmission rounds) under the 3 schemes, where it is easily seen how the total average mean delay is reduced by our time-varying weights algorithm.

Finally, we would like to point out that our time-varying weight assignment algorithm implicitly provides us with the opportunity of tracking the transmission rate of the sources. Fig. 5 illustrates such

	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

Table 1. Mean Delay Per Packet (in Transmission Rounds)

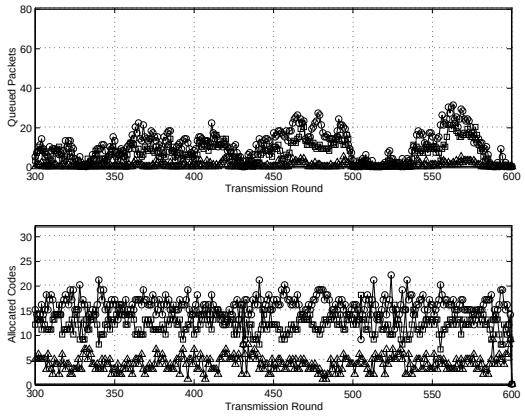

Fig. 4. Time varying ϕ's, Weighted Queues

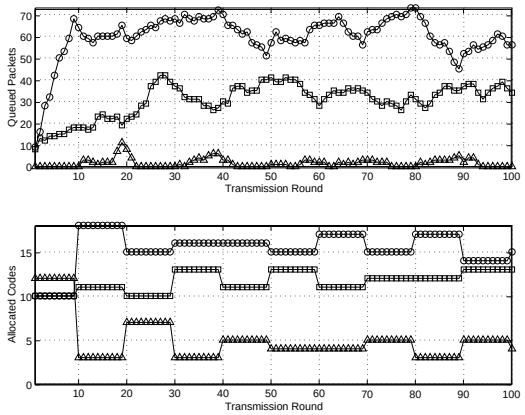

Fig. 5. Tracking input rates

an example. We assume that mobile user m generates traffic which is the sum of a constant bit rate source with rate $\lambda_m/2$ and a Poisson source with rate $\lambda_m/2$ (the λ's are set as in the previous example). Furthermore, we assume that initially we do not know the λ's, and the base-station starts with the arbitrary code assignment of 10,10, and 12 codes to the three users. The base-station updates the ϕ's every 10 transmission rounds, and it can been from Fig. 5 than eventually users 1,2,3 are assigned respectively (on the average) 16,12,4 codes, as it was expected.

5. CONCLUSIONS

In this paper we have presented a dynamic programming algorithm which solves the problem of bandwidth allocation in fair queuing wireless networks with an underlying multicode CDMA physical layer. Our approach is based on a time-varying weight assignment, which minimizes the queuing 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. Future research includes design of distributed algorithms for bandwidth allocation in *ad hoc* wireless networks.

6. 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 Transactions on Networking*, vol. 1, no. 3, 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. of INFOCOM'2000*, Tel Aviv, Israel, March 2000, pp. 1067–1076.

[3] Z.-L. Zhang, D. Towsley, and J. Kurose, "Statistical Analysis of Generalized Processor Sharing Scheduling Discipline," *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 6, pp. 1071–1080, August 1995.

[4] A. Demers, S. Keshav, and S. Shenker, "Analysis and simulation of a fair queueing algorithm," in *Proc. ACM Sigcomm'89*, 1989, pp. 1–12.

[5] R. L. Cruz, "A calculus for network delay, part I: network elements in isolation," *IEEE Transactions on Information Theory*, vol. 37, no. 1, pp. 114–131, January 1991.

[6] H. Zhang, "Service disciplines for guaranteed performance service in packet-switching networks," *Proceedings of the IEEE*, vol. 83, no. 10, pp. 1374–96, October 1995.

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

[8] A. Elwalid and D. Mitra, "Design of Generalized Processor Sharing Schedulers Which Statistically Multiplex Heterogeneous QoS Classes," in *Proc. of INFOCOM'99*, Piscataway, NJ, USA, 1999, pp. 1220–1230.

[9] S. Verdú, *Multiuser Detection*, Cambridge University Press, 1998.

[10] S. J. Golestani, "A self-clocked fair queueing scheme for broadband applications," in *Proc. IEEE Infocom '94*, 1994, pp. 636–646.

[11] P. Goyal, H. M. Vin, and H. Cheng, "Start-time fair queueing: a scheduling algorithm for integrated services packet switching networks," *IEEE Transactions on Networking*, vol. 5, no. 5, pp. 690–704, October 1997.

[12] D. Saha, S. Mukherjee, and S. K. Tripathi, "Carry-over round robin: a simple cell scheduling mechanism for ATM networks," *IEEE Transactions on Networking*, vol. 6, no. 6, pp. 779–796, December 1998.

[13] S. Shreedhar and G. Varghese, "Efficient fair queueing using deficit round robin," *IEEE Transactions on Networking*, vol. 4, no. 3, pp. 375–385, June 1996.

[14] D. Stiliadis and A. Varma, "Efficient Fair Queueing Algorithms for Packet Switched Networks," *IEEE Transactions on Networking*, vol. 6, no. 2, pp. 175–185, April 1998.

[15] J.C. Chen, K.M. Sivalingam, and R. Acharya, "Comparative analysis of wireless atm channel access protocols supporting multimedia traffic," in *Mobile Networks & Applications, Vol. 3, No. 3*, Baltzer; ACM Press, Netherlands, 1998, pp. 293–306.

[16] R. Bellman, *Applied Dynamic Programming*, Princeton Univ., 1962.

[17] N.D. Sidiropoulos and R. Bro, "Mathematical programming algorithms for regression-based nonlinear filtering in $\mathbb{R}^N$," *IEEE Transactions on Signal Processing*, vol. 47, no. 3, pp. 771–782, March 1999.

[18] A. Stamoulis and G.B. Giannakis, "Deterministic Time-Varying Packet Fair Queuing for Integrated Services Networks," in *Proceedings of Globecom2000*, San Franscisco, California, 2000, pp. 621–625.

[19] 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. of INFOCOM'99*, Piscataway, NJ, USA, 1999, pp. 398–404.