Scheduling Algorithms for a Mixture of Real-Time and Non-Real-Time Data in HDR

Sanjay Shakkottai[a] * Alexander L. Stolyar[b]

[a] Coordinated Science Laboratory and
Department of Electrical and Computer Engineering
University of Illinois at Urbana-Champaign

[b] Mathematical Sciences Research Center
Bell Laboratories, Lucent Technologies
600 Mountain Avenue, Murray Hill, NJ 07974

High Data Rate (HDR) technology has recently been proposed as an overlay to CDMA as a means of providing packet data service to mobile users. In this paper, we study various scheduling algorithms for a mixture of real-time and non-real-time data over HDR/CDMA and compare their performance. We study the performance with respect to packet delays and also average throughput, where we use a token based mechanism to give minimum throughput guarantees. We find that a rule which we call the exponential rule performs well with regard to both these criteria. (In a companion paper, we show that this rule is throughput-optimal, i.e., it makes the queues stable if it is feasible to do so with any other scheduling rule.) Our main conclusion is that intelligent scheduling algorithms in conjunction with token based rate control provide an efficient framework for supporting a mixture of real-time and non-real-time data applications in a single carrier.

1. INTRODUCTION

There has been rapid growth of wireless telephony in the past few years. In conjunction with the explosive growth of the Internet, this has led to an increasing demand for wireless data services. In response to this demand, various mechanisms have been proposed to support data traffic over wireless telephony services. One of the proposed schemes to provide high-speed downstream data access over CDMA is High Date Rate (HDR, see [2]).

In this paper, we wish to address the following problems, namely,

(i) How can multiple *real-time* data users be supported simultaneously with good quality of service (QoS) for all users, namely, with packet delays not exceeding given thresholds with high probability.

*This work was carried out when this author was an intern at Bell Labs, Lucent Tech., Murray Hill, NJ 07974

(ii) How can a *mixture* of real-time and non-real-time users be supported simultaneously with real-time users receiving their desired QoS and non-real-time users receiving the maximum possible throughput without compromising the QoS requirements of real-time users.

Wireless scheduling has two peculiarities which distinguish it from conventional wireline scheduling. These are

(i) The channel here is not perfect and is subject to errors. This causes *bursts of errors* to occur during which packets cannot be successfully transmitted on link. The implication of this is that good scheduling algorithms need to be channel quality (state) dependent.

(ii) Channel state varies randomly in time on both slow and fast time scale.

(a) Fast channel variations (due to fast fading) are such that states of different channels can asynchronously switch from "good" to "bad" within a few milliseconds and vice-versa. A good scheduling algorithm should take advantage of this by giving some preference to a user whose channel is currently good.

(b) Slow channel variation means that the average channel state condition depends on user location and interference level. Therefore some users inherently demand more air-interface resources than others, even if their data rate requirement is the same.

The study in this paper compares various wireless scheduling algorithms which explicitly use the above information. Our model and the form of QoS for real-time users are essentially same as those in [1], where the *Modified Largest Weighted Delay First* (M-LWDF) rule was proposed and studied. The most important contribution of this paper is the detailed study of the *Exponential* rule and the gains achieved by the use of this algorithm. (In [12], we also formally show that the exponential rule is *throughput-optimal*, in the sense that it makes the queues stable if it is feasible to do so with any other scheduling rule.) We also study means by which users can be *guaranteed* a minimum throughput, namely employing virtual token queues for this purpose. Guaranteed minimum throughput is a QoS notion appropriate for non-real-time users.

1.1. Related Work

In [16], optimal scheduling for a wireless system consisting on N queues and a single server is studied. The arrival process to each queue are assumed to be and i.i.d. Bernoulli processes. The channel perceived by each queue is also assumed to be an i.i.d. Bernoulli process. The authors show that the policy which minimizes the total number of packets in the system in a stochastic ordering sense is the one which serves the longest queue. However, if the channels are correlated (as is the case in Rayleigh fading channels), then it is clear that a channel state independent policy like this one will perform badly.

In [13], the authors study the problem of scheduling multiple real-time streams with deadlines, over a shared channel. The channel is considered to be ON-OFF. It is observed that a channel state dependent version of the earliest deadline date (EDD) policy is not always optimal in the sense of minimizing the number of packets lost due to deadline expiry. However, for most values of the channel parameters that are of practical interest, it is shown that the modified-EDD is optimal. Recent work in [18] study a dynamic program approach to downlink scheduling, where for various amounts of channel information, the authors characterize policies which maximize a discounted cost using a dynamic program.

The authors in [3] study the effect of the wireless link on the performance of transport protocols such as TCP for various scheduling protocols using a simulation-based approach. They conclude that a channel-state dependent scheduler can lead to significant improvement in channel utilization for typical wireless LAN configurations. This idea is further explored in [7] in the context of fair queueing.

The rest of this paper is as follows. In Section 2, we describe the system model and discuss the simulation parameters. We then describe various scheduling algorithms (in Section 2.1) under consideration and the performance metrics (in Section 2.3) that are used. Finally, in Section 3, we present the simulation results and discuss them.

2. DESCRIPTION OF THE MODEL

A very attractive feature of HDR is that it enables the use of efficient scheduling algorithms. In this section, we describe the CDMA-HDR system model, and describe various algorithms used for scheduling data flows. Two measures of the merit of these algorithms are packet delay distributions and throughput. We discuss these measures and describe a means of ensuring quality of service (QoS) guarantees in conjunction with these measures.

We study a single cell CDMA system with HDR (see [2]). HDR is a downlink packet-data service which occupies a single carrier of a CDMA system. Transmissions for different users are time multiplexed, i.e., data is transmitted to one user at a time at the full power available at the base-station. The cell serves N mobile users, each receiving a data flow[2]. The base-station contains N queues, each corresponding to a different data flow, and an associated scheduler. The scheduling decisions are made once every "time-slot", which is as in HDR, 1.667 m sec.

As mentioned before, all users share a common channel. However, the quality (i.e., the fading level) of the channel seen by different users is different, and changes asynchronously in time. The channel model in our simulation is as follows. Using typical HDR and cell parameters, and using a conservative analysis as in [2], we can derive the average fade level distribution for a typical mobile in a cell. Using this distribution, we pick 14 users, and find the corresponding average data rate to each user as in Table 1. Each entry corresponds to the mean data rate received by the corresponding user *if the base-station dedicated all transmission time to that user*. Then, we assume that the channels are Rayleigh fading with the mean being that chosen above. We next note that HDR does not support arbitrary transmission rates. In our simulations, we assume that the actual set of available rates is 2^i*9.6 kbps, $i = 0, 1, 2, \ldots$[3]. The actual transmission rate is chosen to be the largest available rate such that $\frac{E_b}{N_0}$ is satisfied. We denote the actual maximum rate that the channel can support over a slot as the *state of the channel* at that slot. Finally, in our simulations, we assume that the speed of the mobiles are 6 mph.

[2]This means that the packets of each flow must be delivered in order.

[3]The exact rate set in HDR is given in [2].

Table 1
Mean throughput supported by the Rayleigh fading channels.

User Number	"Mean" Rate ($\bar{\mu}_i$ in kbps)
1	398.72
2	397.14
3	397.14
4	384.91
5	360.88
6	342.00
7	276.01
8	241.43
9	232.18
10	220.60
11	191.24
12	154.27
13	150.72
14	137.80

This corresponds to a doppler frequency of 8 Hz, roughly characterizing how "fast" the variations of the channel quality are.

The scheduler makes a decision to serve a particular queue at the beginning of every time slot. This decision could depend on packet delays and channel states. We assume that the scheduler has (current or delayed) channel state information and also the head of line (HOL) packet delays. Once a decision has been made, the chosen queue is serviced in the slot at the rate corresponding to the state of its channel.

In the simulations, we assume that $N = 14$. Each of these users require an average rate of 28.8 kbps. This corresponds to the typical rate required for streaming audio over the Internet. The arrival processes to each of the queues are Bernoulli processes with a mean rate of 28.8kbps, with a packet size being 128 bytes (corresponding to a HDR packet). Real-time users like streaming audio will indeed generate a smooth traffic, and hence, a Bernoulli model seems reasonable for such traffic. On the other hand, in our study, a non-real-time or a very bursty traffic is modeled by its "worst case", an infinite amount of data to send.

2.1. Scheduling Algorithms

The scheduling algorithms under consideration are the FIFO rule, maximum rate rule (MAX-RATE), the modified largest weighted delay first rule (M-LWDF), the proportionally fair rule (PROP-FAIR), and the exponential rule (EXP).
Let us define

$\mu_i(t)$ to be the state of the channel of user i at time t, i.e., the actual rate supported by the channel. This rate is constant over one slot.

$\bar{\mu}_i$ to be the rate corresponding to the mean fading level of user i as in Table 1.

$W_i(t)$ to be the amount of time the HOL packet of user i has spent at the base-station.

The FIFO discipline schedules the user whose HOL packet has spent the longest time at the base-station. Formally, we schedule at time t, the user

$$j = \arg\max_i W_i(t)$$

This policy is oblivious to the channel state, and as can be expected, performs very poorly. In the results presented in Section 3, we do not include the performance of this policy because for typical loads supported by other policies, this policy renders the system unstable.

The maximum rate rule schedules the user whose channel can support the largest data rate over the next slot, i.e.,

$$j = \arg\max_i \mu_i(t)$$

This clearly requires channel state information. We study the case where the exact channel state is known and also the case when only delayed channel state information is available.

The modified largest weighted delay first (M-LWDF) rule has been proposed in [1]. For any arbitrary set of $\gamma_i > 0$, the rule is given by

$$j = \arg\max_i \gamma_i \; \mu_i(t) \; W_i(t)$$

It has been proven in [1] that this rule is *throughput optimal.* (A policy is said to be throughput-optimal if it satisfies the property that it renders the queues at the base-station stable if any other rule can do so. In other words, it has the largest stable admission region.) It was also demonstrated by simulation that a "good" choice of γ_i is given by $\gamma_i = \frac{a_i}{\bar{\mu}_i}$, where $a_i > 0, i = 1, \ldots, N$, are suitable weights, which characterize the desired QoS. This rule tries to balance the weighted delays of packets and also tries to utilize the channels in a good manner.

The proportionally fair rule (PROP-FAIR) has been proposed for HDR (see [17,8]). As the name suggests, this rule attempts a "proportionally fair" allocation of rates to different users. We consider a version of this rule, given by

$$j = \arg\max_i \frac{\mu_i(t)}{\bar{\mu}_i} \tag{1}$$

Remark. The above rule (1) is not the "true" proportionally fair rule as in [17,8]. The true proportionally fair rule is given by

$$j = \arg\max_i \frac{\mu_i(t)}{\tilde{\mu}_i} \tag{2}$$

where $\tilde{\mu}_i$ is the mean rate actually given to user i, and measured over a certain (relatively long) "sliding window" (see [17,8]). In the case when all users have infinite amount of data to send to (and all users have the same model of channel variations but with different average levels, as in our simulations), the rules (1) and (2) are approximately equivalent. We choose the version (1) for our simulations, because in the case of data flows arriving at finite rates, for some users we have $\tilde{\mu}_i = \lambda_i$ (i.e. the entire flow "goes through") no matter how good their average channel condition is; as a result, for those users, $\tilde{\mu}_i$ is not a good measure of the actual "amount of the air interface resource" allocated to them.

We note that the PROP-FAIR rule (any version!) does not account for delays of packets and can result in poor delay performance (see Section 2.3). Further, it can be easily shown that this rule is *not throughput-optimal.*

Study of the *exponential* rule, briefly introduced as a heuristics in [1], is in the center of this work, so it is discussed in more detail in the next Section.

2.2. The Exponential Rule

Let us fix any set of positive constants $\gamma_i > 0$ and $a_i > 0$, $i = 1, 2, \ldots N$. Then, the *exponential* rule is given by

$$j \;=\; \arg\max_i \gamma_i \; \mu_i(t) \;\; \exp\left(\frac{a_i W_i(t) - \overline{aW}}{1 + \sqrt{\overline{aW}}}\right)$$

where $\overline{aW} = \frac{1}{N}\sum_i a_i W_i(t)$.

2.2.1. Intuition

For "reasonable" values of γ_i and a_i (this issue is discussed in Section 2.2.3), this policy tries to equalize the weighted delays $a_i W_i(t)$ of all the queues *when their differences are large.* As we can see, if one of the queues would have a larger (weighted) delay than the others by more than order $\sqrt{\overline{aW}}$, then the exponent term becomes very large and overrides channel considerations (as long as its channel can support a non-zero rate), hence leading to that queue getting priority. (We note that the $\overline{aW}$ term in the exponent can be dropped without changing the rule as it is common for all queues. This is present only to emphasize the motivation for this rule.) On the other hand, for small weighted delay differences (i.e., less than order $\sqrt{\overline{aW}}$), the exponential term is close to 1 and the policy becomes the proportionally fair rule. Hence, this policy *gracefully adapts* from a proportionally fair one to one which balances delays. The factor 1 in the denominator of the rule is present simply to prevent the exponent from blowing up when the weighted delays are small. We discuss the merits of this rule in greater detail in Section 3.

2.2.2. Throughput-Optimality

In our companion paper [12], we prove that the exponential rule is throughput optimal. (We remind, this means that it renders the queues at the base station stable if any other rule can do so.) The assumptions on the channel and arrival processes are quite general. We assume that the *aggregate* channel process (including states of all individual channels), is Markovian with a finite number of states. Note that the *individual* channels need not be independent. The aggregate input process is assumed to be an ergodic (discrete time) Markov chain.

The proof of the throughput-optimality uses *fluid limit* technique [10,5,4,14,6], along with a *separation of time-scales* argument. We refer reader to [12] for the details.

2.2.3. Choice of Parameters

As in M-LWDF, the choice of γ_i is given by $\gamma_i = \frac{a_i}{\bar{\mu}_i}$. The values of $a_i, i = 1 \ldots N$, determine the tradeoff between reducing weighted delays and being proportionally fair when delays are small. The QoS requirement would be of the form

$$\mathrm{Pr}\left(\mathrm{W_i} > \mathrm{T_i}\right) \quad \leq \quad \delta_i$$

where W_i is delay encountered by a typical packet of user i, T_i is the maximum delay that user i can tolerate (measured in slots) and δ_i is the largest probability with which the system is allowed to violate the delay requirement. For the system we are interested in, a typical requirement would be a violation probability of 10^{-2} corresponding to a maximum delay of the order of a second. As play-out buffers for streaming audio over the Internet are sufficiently large, the above values are a reasonable requirement.

A rule of thumb for choosing a_i which works in practise is $a_i = \frac{-\log(\delta_i)}{T_i}$. This rule is suggested by large deviation optimality results of [15]. We note that the performance of the rule is robust with respect to moderate changes of T_i and δ_i. Simulation results exploring this aspect are available in [11].

2.3. Performance Metrics

In this section, we discuss the performance metrics we employ to evaluate the scheduling algorithms. These are the delay criterion and the average throughput. One or the other is more important depending on the type of the traffic.

For real-time traffic, a good measure of performance is the delays packets incur at the base-station. We assume for simplicity that all users need the same delay QoS, i.e., they all have the same T_i and δ_i. A good scheduling algorithm should keep all delays below T_i with high probability, which is achieved roughly when the delays are kept close for all users. In the setup we study, the worst user (the mobile with the worst channel) has the average channel quality much worse than the best one. From Table 1, the best user can support a mean rate of 398.72 kbps (if it would be the only user of the cell) while the worst one supports only 137.8 kbps. However, we want to support 14 users simultaneously, each with a mean rate 28.8 kbps, leading to a total rate of 403.2 kbps. A naive scheduling algorithm which *does not* depend on the channel state will find it impossible to achieve this objective. However, a smart algorithm which favors users with relatively good channels would hopefully be able to achieve this objective.

If a system with real-time users only is stable, then clearly, the long term throughput is simply the arrival rate. As we remarked earlier, the M-LWDF and the EXP rule are optimal in this regard, i.e., they can support the largest set of available rates. The proofs of this can be seen in [1,12]. Finally, we comment that scheduling algorithms which keep the delays of all the users about the same and keep them all reasonably small are superior to those which may have better delay tails for one of the users but have very bad delays for other users.

Another important measure is the long term throughput available to each of the queues. For non-real time data, this is the main measure of interest. We would like to give minimum throughput guarantees to these users.

Suppose we have a mixture of coexisting real-time and non-real-time users in the cell. In this case, we want to serve *all* the real-time users with low flow delays *and* provide minimum rate guarantees to non-real-time users. Moreover, we want leftover capacity to be allocated to non-real-time users in a proportionally fair manner. We show that this can be accomplished using token based versions of our scheduling schemes.

2.4. Token Queues for Minimum Throughput Guarantees

The M-LWDF and EXP rules, as they described above, make scheduling decisions based on the actual packet delays (in addition to the channel conditions). If providing a certain minimum throughput to a flow is a goal, those rules can be modified as follows. Suppose, associated with each user, there is a *virtual* token queue into which tokens arrive at a constant rate r_i, the desired throughput of user i. (I.e., a *counter* representing the token queue is incremented by $r_i \Delta t$ at regular time points, Δt time units apart.) Let us define $V_i(t)$ to be the delay of the head of line token in the user i token queue. Note that we do not need to actually maintain the token delays. As the arrival rates of tokens are constant, $V_i(t) = \frac{Q_i(t)}{r_i}$, where $Q_i(t)$ is token queue length (a counter value) at time t.

Then, we use the M-LWDF and EXP rules with $W_i(t)$ being replaced by $V_i(t)$. After the service of a (real) queue, the number of tokens in the corresponding token queue is reduced by the actual amount of data transmitted. ("Pathological" cases are treated as follows. If a user is scheduled, and does not have enough packets to send, the token queue is depleted by the amount *as if* it had enough data to send. If user should be scheduled, but has *no* data to send, its token queue is depleted by the corresponding amount, and scheduler chooses the "second best" queue to serve, and so on. If a token queue is shorter than the number of tokens that needs to be removed, it is set to 0.) To increase the algorithm's robustness, for real-time users, we will set the token rates to be slightly larger (by 2%) than the desired throughput.

A token queue is similar to a leaky bucket regulator followed by a real queue based scheduler. However, there are some differences. With a good scheduling algorithm, a token queue mechanism can allow a user to utilize *more* than the required minimum rate if all the token queues are small. Such an effect is possible with leaky bucket regulators if the regulator marks packets as high and low priority and the scheduling algorithm operates on the high priority queue lengths and packet delays. However, this could lead to out sequence delivery for the low priority packets. This would not be so with the token queue mechanism.

3. SIMULATION RESULTS AND DISCUSSION

In this section, we present simulation results for the system described in the previous section. As described earlier, we consider a CDMA/HDR cell with 14 users. The channels are Rayleigh fading, which have been implemented using Jakes's implementation of Clarke's fading model.

3.1. Real Queue based Scheduling

In Figure 1, we plot the delay distribution tails for the best and worst users(i.e., users 1 and 14 in Table 1). The arrival process to each queue is a Bernoulli process with a mean rate of 28.8 kbps. We assume that the scheduler has knowledge of the current state of the channel. We observe that for the best user, the MAX-RATE rule provides the best delay distribution tails. However, for the worst user, the queue becomes unstable, leading to $\mathrm{Pr(W_i > d) = 1}$ for all $d > 0$. This can be understood from the fact that at each slot, this algorithm schedules the queue whose channel can support the maximum rate. Therefore, most of the time, this leads to the user corresponding to the better channel utilizing a slot whenever it has packets. However, this greedy scheme leads to suboptimal utilization

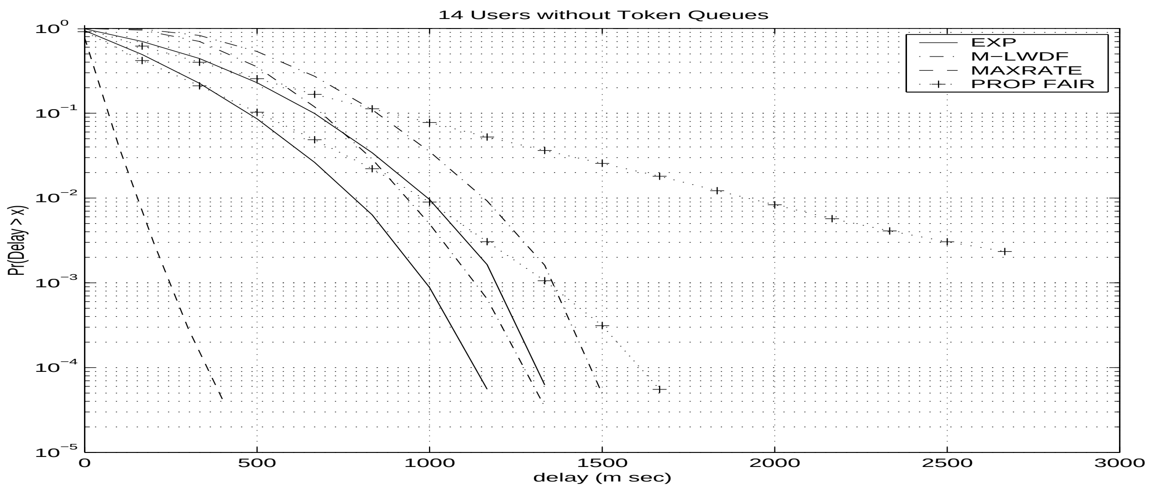

Figure 1. Real queue based scheduling. Data rate: 28.8 kbps per user. Delay tails of the best and worst users are plotted. The scheduler has exact channel knowledge.

of the system[4] leading to instability for users with poorer channels.

As can be observed from these plots, the exponential rule leads to the best performance, keeping the delays of the best and worst users close, as well as providing the best performance among all the stable algorithms. On the other hand, as mentioned earlier, the PROP-FAIR rule does not lead good delay performance, with the performance being an order of magnitude worse than the EXP rule at the desired QoS. Qualitatively similar results hold true even when only delayed channel information is available to the scheduler (see [11]).

3.2. Token Queues Based Scheduling for a Mixture of Real-Time and Non-Real-Time Users

We next study the case when we have both real-time and non-real-time users. A non-real-time user is modeled as a user sending at rate much greater than 28.8 kbps. Token queues are used to guarantee a minimum throughput of 28.8 kbps to all users, real-time as well as non-real-time. In this section, we do not study the MAX RATE algorithm. This is because even with real-time users only, the algorithm made queues unstable as seen from Figure 1.

In Figures 2 and 3, we study the throughput and delays for a system consisting of 14 users. User 1 is a non-real-time user, sending packets at a rate of 288 kbps. The queue corresponding to user 1 is back-logged all the time, hence, the delays of the best and worst *real-time* users are plotted, i.e., users 2 and 14 respectively.

We note that all three algorithms achieve the desired minimum throughput. However, the EXP and the PROP FAIR rules allocate a larger rate than M-LWDF for the non-real-time user, without compromising the stability of real-time users' queues. Meanwhile, the QoS of the real-time users (delay performance) is much better for the EXP rule as

[4]It follows from the analysis in [1] that this scheme is not throughput-optimal.

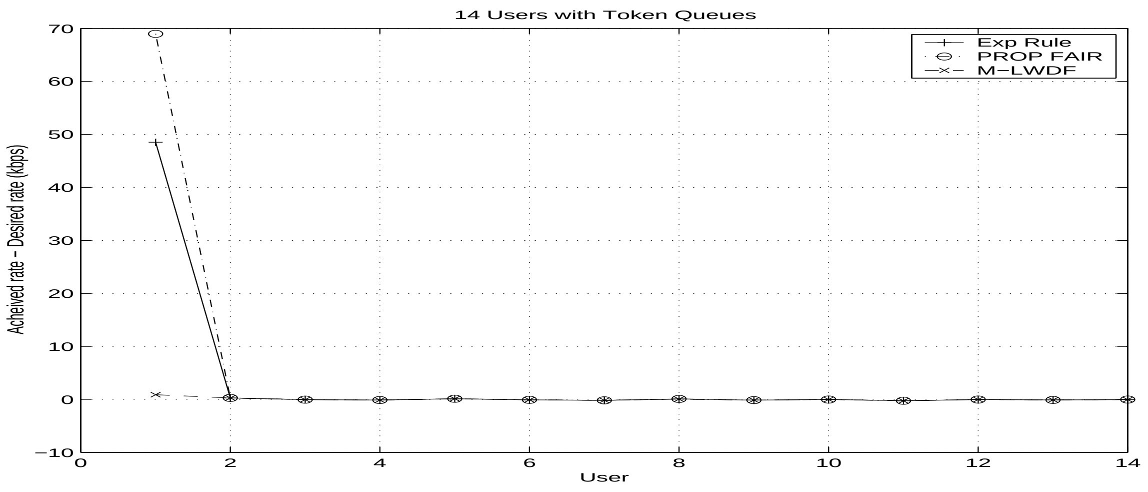

Figure 2. Token Queue based scheduling. Data rate: 28.8 kbps per user. User 1 has infinite backlog.

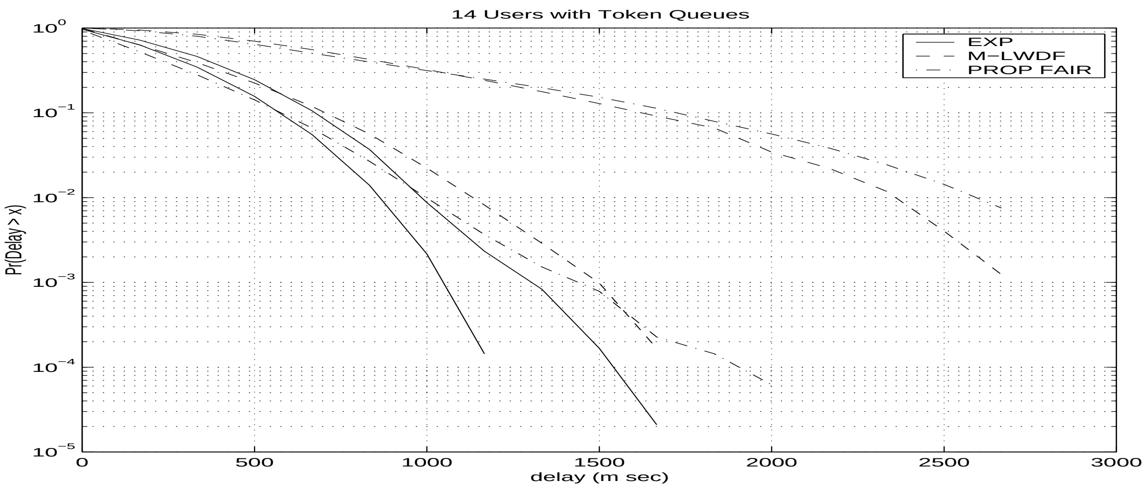

Figure 3. Token Queue based scheduling. Data rate: 28.8 kbps per user. User 1 has infinite backlog. The delay distribution tails of users 2 and 14 are plotted.

compared to the other rules. This observation can be explained by the fact that when token queues are small, the EXP rule is basically the same as the PROP FAIR rule. However, when delays start becoming unbalanced, the exponential term balances the delays. Further simulation results for other systems (see [11]) also bear out our previous observations. These results indicate that we can indeed support 14 users with the required rates and hence, as discussed in Section 2.3, supporting more than any single users' average channel throughput!

4. CONCLUSIONS

In this paper, we have compared various scheduling algorithms to support a mixture of real-time and non-real-time data in CDMA/HDR. We defined two performance metrics, namely, packet delays and guaranteed throughput. Throughput guarantees were implemented using a token queue mechanism. With respect to either these measures we found that the EXP rule performs the best among the algorithms considered. We also investigated the robustness of the EXP rule to the choice of its parameters. Our results show that an intelligent scheduling algorithm, in conjunction with a token queue mechanism allows us to support a mixture of real-time and non-real-time data over HDR with high efficiency.

REFERENCES

1. M. Andrews, K. Kumaran, K. Ramanan, A. L. Stolyar, R. Vijayakumar, P. Whiting "CDMA Data QoS Scheduling on the Forward Link with Variable Channel Conditions," Bell Laboratories Technical Report, April, 2000.
2. P.Bender, P.Black, M.Grob, R.Padovani, N.Sindhushayana, A.Viterbi, "CDMA/HDR: A Bandwidth Efficient High Speed Wireless Data Service for Nomadic Users," *IEEE Communications Magazine*, July 2000.
3. P. Bhagwat, P. Bhattacharya, A. Krishna and S. Tripathi, "Enhancing throughput over wireless LANs using channel state dependent packet scheduling," *Proc. IEEE Infocom'97*, April 1997.
4. H. Chen, "Fluid Approximations and Stability of Multiclass Queueing Networks: Work-conserving Disciplines," *Annals of Applied Probability*, Vol. 5, (1995), pp. 637-665.
5. J. G. Dai, "On the Positive Harris Recurrence for Open Multiclass Queueing Networks: A Unified Approach Via Fluid Limit Models," *Annals of Applied Probability*, Vol. 5, (1995), pp. 49-77.
6. J. G. Dai and S. P. Meyn, "Stability and Convergence of Moments for Open Multiclass Queueing Networks Via Fluid Limit Models," *IEEE Transactions on Automatic Control*, Vol. 40, (1995), pp. 1889-1904.
7. C. Fragouli, V. Sivaraman and M. Srivastava, "Controlled multimedia wireless link sharing via enhanced class-based queueing with channel-state-dependent packet scheduling," *Proc. IEEE Infocom'98*, April 1998.
8. A. Jalali, R. Padovani, R. Pankaj, "Data Throughput of CDMA-HDR a High Efficiency - High Data Rate Personal Communication Wireless System," *Proc. VTC-2000-Spring.*
9. F. Kelly, "Charging and Rate Control for Elastic Traffic," *European Transactions on Telecommunication*, Vol. 8, 1997, pp. 33-37.
10. A.N. Rybko and A.L. Stolyar, "Ergodicity of Stochastic Processes Describing the Operation of Open Queueing Networks," *Problems of Information Transmission*, Vol. 28, (1992), pp. 199-220.
11. S. Shakkottai and A. L. Stolyar, "A Study of Scheduling Algorithms for a Mixture of Real- and Non-Real-Time Data in HDR," Bell Laboratories Technical Report, October, 2000.

12. S. Shakkottai and A. L. Stolyar, "Scheduling for Multiple Flows Sharing a Time-Varying Channel: The Exponential Rule," Bell Laboratories Technical Report, December, 2000. Submitted.
13. S. Shakkottai and R. Srikant, "Scheduling Real-Time Traffic With Deadlines Over a Wireless Channel," *Proc. ACM Workshop on Wireless and Mobile Multimedia*, Seattle, WA, August 1999.
14. A.L. Stolyar, "On the Stability of Multiclass Queueing Networks: A Relaxed Sufficient Condition via Limiting Fluid Processes," *Markov Processes and Related Fields*, 1(4), 1995, pp.491-512.
15. A. L. Stolyar and K. Ramanan, "Largest Weighted Delay First Scheduling: Large Deviations and Optimality," *Annals of Applied Probability*, Vol. 11, (2001), No.1.
16. L. Tassiulas and A. Ephremides, "Dynamic Server Allocation to Parallel Queue with Randomly Varying Connectivity," *IEEE Tran. Info. Theory*, Vol. 30, No. 2, March 1993.
17. D. Tse, "Forward Link Multiuser Diversity Through Rate Adaptation and Scheduling," Submitted to *IEEE JSAC*.
18. D. Zhang and K. M. Wasserman, "Stochastic transmission policies for time-varying channels with memory under incomplete channel state information," *Proc. 38th Annual Allerton Conf. Commun., Control, and Comput.*, University of Illinois, Urbana, IL, 2000.