Performance Tradeoffs in Wireless Scheduling with Flow Aggregration

Raphael Rom and Hwee Pink Tan
Department of Electrical Engineering
Technion, Israel Institute of Technology
E-mail: rom@ee.technion.ac.il and hweepink@tx.technion.ac.il

***Abstract*— We develop Markov models for various schedulers in order to evaluate their performance tradeoffs at wireless links. While FIFO scheduling aggregates flows into a single flow just prior to the wireless link, Channel-State Dependent (or CSD) schedulers maintain a queue for each flow, and use predicted channel information to make scheduling decisions. We present results of tradeoffs in terms of overall throughput and per-flow QoS performance obtained with each scheduler under different channel conditions.**

I. Introduction

We define a *flow* to be a sequence of packets that traverses from one network element to the next while preserving the packet ordering. In flow aggregation, multiple flows (known as *constituent flows*) are combined into a single aggregate flow. Once the aggregation is performed, the remaining network elements along the path simply regard the aggregate flow as a *single* flow whose rate is the sum of the reserved rates of the constituent flows. An example of an application of flow aggregation is in the Class-Based QoS Framework proposed in [1], wherein each class offers characteristic performance to its users. Hence, individual flows with similar QoS requirements and common source-destination specifications can be aggregated into a single flow. In this way, buffer management can be simplified and scheduling efficiency can be improved. In addition, rerouting an existing aggregate flow in the event of link failure is much more efficient than rerouting each of the individual constituent flows.

However, let us consider the scenario where *N* flows are aggregated at a common source node and are destined to *N* mobile users linked by a common base station, as depicted in Fig. 1.

Since the channel characteristics of the wireless link are spatially and time dependent, at any instant, the channel to user *i* may be good while that to user *j* may be error-prone.

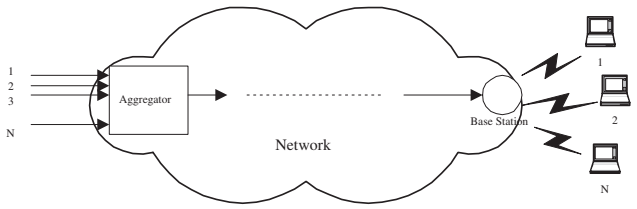


Fig. 1. Flow Aggregation at Wireless Link

With flow aggregation, packets destined for different users are served by a single queue (*FIFO Scheduler*). As a result, we have the problem of *Head-of-Line* (HOL) blocking, where a HOL packet will be repeatedly polled if the channel to its destination is under an error burst. As such, the wireless channel is not utilized efficiently, and this may result in degradation in overall as well as per-flow QoS performance.

The work in [2] illustrated that in the scenario depicted in Fig. 1, when the wireless channel is in burst error state, FIFO packet scheduling at the base station causes poor utilization of the wireless channel and unfair sharing of bandwidth. As a result, a class of scheduling algorithms were developed which explicitly takes into consideration the wireless channel characteristics when making packet dispatching decisions (*Channel State Dependent or CSD Schedulers*). Simulation results indicate significant improvement over FIFO scheduling in terms of overall throughput at the transport layer.

In this study, we develop Markov models for the schedulers described above in order to investigate their performance tradeoff at a wireless link. We consider a N-flow scheduling problem at the base station, where equal-sized packets of each flow are destined to *N* different mobile users. The scheduler allocates fixed-size time slots (equal to packet transmission time) to flow *j* in order to meet its reserved rate r_j, where $r_j \epsilon Z+$. We assume that all the input queues have infinite capacity, and the arrival rates to each queue are such that all queues are always backlogged. In addition, we also assume that the scheduler performs infinite retransmission, i.e., no packets are dropped.

For the wireless channel, we consider a Two-State Markov Chain error model where in each slot *i*, the channel of each flow *j* can be in one of two states $\{Good, Bad\}$. Transmissions that occur when the channel of a flow is in *Good* state is always successful, while that in a *Bad* state is always unsuccessful.

II. FIFO scheduler

We define our FIFO scheduler to comprise a fair aggregator followed by a packet dispatcher. The fair aggregator schedules packets from each flow according to r_j such that they arrive in a single queue at the packet dispatcher as depicted in Fig. 2.

The above mechanism is equivalent to a Weighted Round Robin (WRR) scheduler that allocates slots to flow *j* until r_j packets have been *transmitted* before switching to flow

j+1 and so on. In other words, the FIFO scheduler performs 'repeated' polling in order to transmit packets according to the transmission cycle depicted in Fig. 2.

Let us define A_i such that if $A_i = Sj,\ 1 \leq j \leq N$, then slot *i* is allocated to flow *j* and only flow *j* is allowed to transmit in this slot. In order to implement the slot allocation policy of the FIFO scheduler, we define the variable $count_i$ such that *count* is incremented at the end of each slot whenever a successful packet transmission takes place, and is reset to zero at the beginning of each transmission cycle, i.e.,

$$count_i = \begin{cases} count_{i-1} + 1, & \text{transmission in slot i;} \\ count_{i-1}, & \text{otherwise.} \end{cases} \quad (1)$$

If *count* is computed modulo *R*, where $R = \sum_{j=1}^{N} r_j$, then the slot allocation mechanism can be defined in terms of count as follows:

$$A_i = \begin{cases} S1, & 0 \leq count_{i-1} \leq r_1 - 1; \\ Sj & \sum_{k=1}^{j-1} r_k \leq count_{i-1} \leq \sum_{k=1}^{j} r_k - 1, \\ & 2 \leq j \leq N. \end{cases} \quad (2)$$

III. Channel-State Dependent (CSD) Scheduler

We define our CSD scheduler to comprise a WRR scheduler, a Channel Status Monitor (CSM) as well as a packet dispatcher. At the beginning of slot *i*, the WRR scheduler allocates the transmission priority in slot *i* based on the reserved rates, r_j, of each flow. The CSM predicts the channel status of each flow in slot *i*. Based on these information, the packet dispatcher selects a flow and transmits its HOL packet. The components of the CSD scheduler are depicted in Fig. 3.

A. WRR Scheduler

In CSD scheduling, slots are allocated in a WRR manner according to the sequence depicted in Fig. 4.

To characterize the WRR scheduler, we define the variable $count_i$ such that *count* is incremented at the end of each slot, and is reset to zero at the beginning of each allocation cycle. The slot allocation mechanism can then be defined in terms of count as given in Eq. (2).

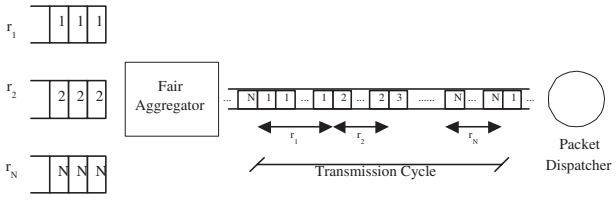


Fig. 2. Mechanism of FIFO Scheduler

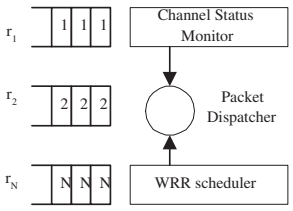


Fig. 3. Components of Channel-State Dependent(CSD) Scheduler

B. CSM

Since every mobile user can hear the transmission from the base station, we consider a slight variation of the CSMA/CA with ACK scheme recommended for IEEE 802.11 where an ACK is returned by *every* station immediately after hearing the transmission from the base station. We assume that the size of the ACK packet is significantly smaller than a slot, and that every ACK packet that is transmitted is always received successfully by the base station. Hence, at the beginning of slot *i*, the CSM is aware of the channel status of each flow in previous slots, i.e., slots *i-1*,*i-2*...and can use this information to predict the channel status in slot *i*.

C. Packet Dispatcher

The packet dispatcher selects a flow for transmission based on the following heuristics: if the CSM predicts a Good channel for the allocated flow, then this flow is selected for transmission; otherwise, if the CSM predicts a Good channel for more than one of the non-allocated flows, an arbitrary flow is selected. We consider three variants of CSD schedulers, namely CSD-ZCK, CSD-PCK and CSD-OSP.

1) CSD-Zero Channel Knowledge (CSD-ZCK): This scheduler corresponds to the CSDP-ZCK scheduler defined in [2]. In this case, scheduling is performed without any channel information, as in the FIFO scheduler. However, since slots are allocated to flows according to allocation cycle depicted in Fig. 4, repeated polling of a flow *j* (as seen in FIFO schedulers) is limited to its reserved rate, r_j. We term this 'blind' polling, since it is performed without using any channel information. Hence, this case will form the lower bound on the achievable throughput of CSD schedulers.

2) CSD-Perfect Channel Knowledge (CSD-PCK): This scheduler corresponds to the CSDP-RR scheduler defined in [2]. In this case, we assume that the CSM performs perfect channel prediction, i.e., at the beginning of slot *i*, the scheduler has perfect knowledge of the channel status of each flow in slot *i*. Although the allocation cycle given by the WRR scheduler determines the transmission priority in each slot, if the allocated flow perceives a Bad channel, another flow that perceives a Good channel can transmit in its place. This eradicates blind polling as seen in CSD-ZCK. Hence, a slot will only be wasted if a flow that perceives a Good channel in the given slot does not exists. This case will form the upper bound on the achievable throughput of CSD schedulers.

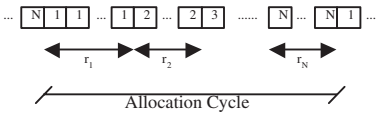


Fig. 4. Illustration of WRR slot allocation

3) CSD-One-Step Prediction (CSD-OSP): It was mentioned in [3] that a simple one-step channel prediction algorithm achieves acceptable performance in CSD scheduling. In one-step prediction, the channel in slot *i* is predicted to be Good if the channel in slot *i-1* is observed to be Good; otherwise, the channel in slot *i* is predicted to be Bad.

Hence, although flow *j* has priority to transmit in a slot allocated to it, if the CSM predicts a Bad channel for this flow, the priority will be given to another flow that predicts a Good channel. If the CSM predicts a Good channel for more than one flow, an arbitrary flow will be selected.

IV. Markov Analysis of Scheduling Mechanisms

Based on Section II, the slot allocation mechanism of FIFO scheduling can be characterized by a single variable, *count*, which tracks the transmission cycle. Given $count_{i-1}$, $A_i = Sj$ can be determined from Eq. (2). Hence, the channel characteristics of flow *j* in slot *i* determines whether a transmission takes place, which in turn determines $count_i$ based on Eq. (1). Hence, we can define a Markov Chain with state variable $count_i$ defined at each slot interval where the state transition probability can be determined given the channel characteristics.

To illustrate, consider a 2-flow homogeneous FIFO scheduler with r_1=r_2=1. The state-transition diagram of the Markov Chain is given in Fig. 5. To evaluate the throughput and delay performance, let us consider the instance of departure of packet *p-1* of flow 1, i_{p-1}, and assume that $count_{i_{p-1}}$ is known. Let us denote the delay of packet *p* as d_p and define the following events:

$$\begin{aligned} fjE &\equiv flow\ j\ in\ error \\ fjC &\equiv flow\ j\ is\ error-free \\ A^k &\equiv Event\ A\ occurs\ in\ k\ consecutive\ slots \\ AB &\equiv Event\ B\ occurs\ after\ Event\ A \end{aligned}$$

We have the following cases:

A. Case A: $count_{i_{p-1}}$=0

In this case, according to Eq. (2), slots will be allocated to flow 1 until its HOL packet (i.e., packet *p*) is transmitted. Hence, we obtain the following expressions:

$$\begin{aligned} Prob(d_p = n) &= f1E^{n-1}f1C \qquad (3) \\ Prob(count_{i_p} = 1) &= \sum_{n=1}^{\infty} Prob(d_p = n) \end{aligned}$$

B. Case B: $count_{i_{p-1}}$=1

In this case, slots are allocated to flow 2 until its HOL packet is transmitted. Subsequent slots will be allocated to flow 1 until packet *p* is transmitted. Hence, we have the following expressions:

$$\begin{aligned} Prob(d_p = n) &= \sum_{m=1}^{n-1} f2E^{m-1}f2Cf1E^{n-1}f1C \\ Prob(count_{i_p} = 1) &= \sum_{n=2}^{\infty} Prob(d_p = n) \qquad (4) \end{aligned}$$

If we define the initial state distribution of the Markov Chain such that $Prob(count_0 = 0) = Prob(count_0 = 1)$ = 0.5, then we can obtain the steady-state delay distribution of flow 1, d_1(n), recursively using Eq. (3) and (4). The distribution for flow 2, d_2(n), under non-homogeneous scheduling can be obtained in a similar manner.

CSD schedulers are analyzed in a similar manner as described above. A similar approach has been used to analyze and evaluate the performance of Wireless-Fair Scheduling [4], and details of the analysis can be found in [5].

V. Performance Evaluation

We evaluate the performance of FIFO scheduling and variants of CSD-scheduling under various channel conditions with homogeneous input flows, i.e., $r_j = 1,\ 1 \le j \le N$, for various N. The channel condition for each flow is specified in terms of the average error rate, P_B, and correlation parameter, p_{corr}. The lower the value of p_{corr}, the more bursty the errors are; p_{corr} = 1 corresponds to the case where the errors are uncorrelated across successive slots. We assume that the parameters P_B and p_{corr} are the same for each flow.

A. Overall Throughput

Based on the delay distribution, $d_j(n)$, derived in Section IV, we can compute the mean packet delay, $\bar{d_j}$, for flow *j*. If we further denote S_j as the throughput of flow *j* and *S* as the overall throughput, then

$$\begin{aligned} S_j &= \frac{1}{\bar{d_j}} \\ S &= \sum_{j=1}^{N} S_j \end{aligned}$$

We plot the values of S for each scheduler for N=2, 3 and 4 in Fig. 6, Fig. 7 and Fig. 8 respectively.

When the channel errors become more correlated, i.e., as p_{corr} decreases, the throughput achievable with the FIFO scheduler decreases dramatically. This is because the HOL packet is continuously polled under an error burst until transmission is successful. Hence, the longer the error burst, the more channel bandwidth is wasted due to repeated polling.

On the other hand, with CSD-ZCK, each HOL packet is polled at most once, and hence, channel bandwidth is lost due only to blind polling. This loss is removed with CSD-PCK, since only flows with Good channels will be polled.

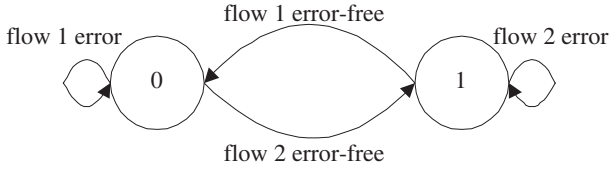
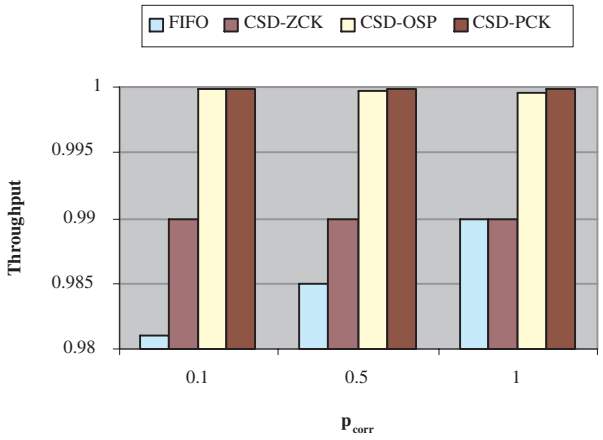
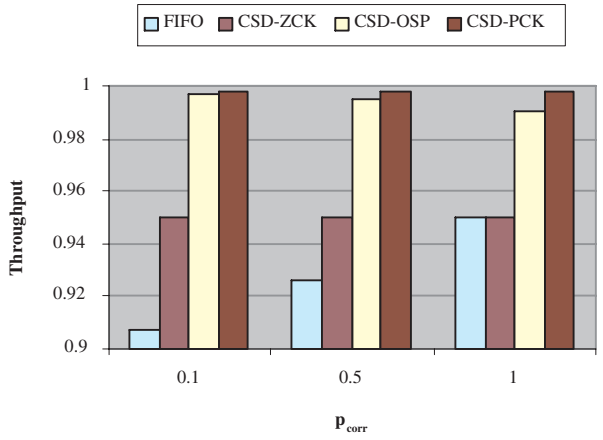


Fig. 5. State transition diagram for 2-flow homogeneous FIFO scheduler

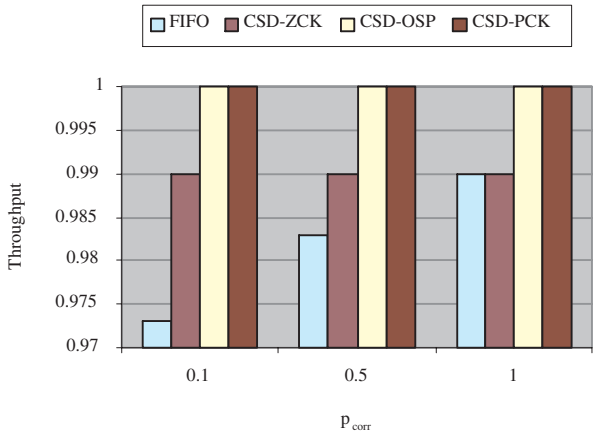


(a)

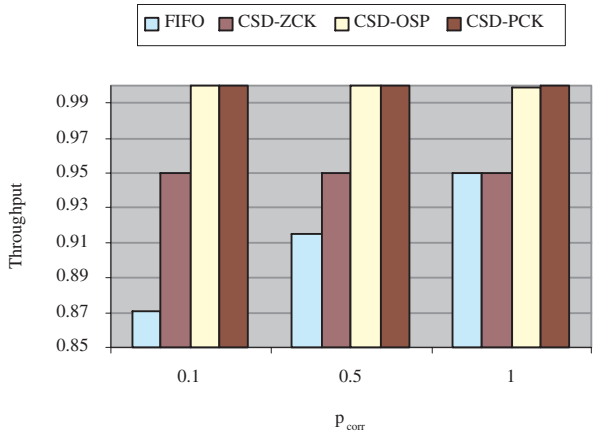


(b)

Fig. 6. Overall throughput for N=2 for (a) P_B=0.01 and (b) P_B=0.05



(a)



(b)

Fig. 7. Overall throughput for N=3 for (a) P_B=0.01 and (b) P_B=0.05

However, when channel errors become uncorrelated (i.e., p_{corr}=1.0), under the given channel conditions, the length of the error burst is computed and found to be within one slot, and hence, no channel bandwidth is wasted due to repeated polling. Hence, FIFO and CSD-ZCK achieves identical throughput since the bandwidth is lost only through blind polling. With perfect channel knowledge, no bandwidth is lost through repeated or blind polling - hence, CSD-PCK achieves the highest throughput amongst all the algorithms.

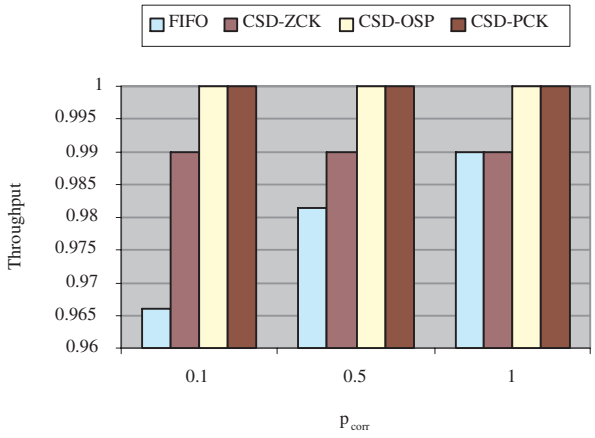
For N=2, the degradation in overall throughput with CSD-OSP (relative to CSD-PCK) is negligible ($< 0.05\%$ for P_B=0.05, p_{corr}=0.1) when the channel is highly correlated, and increases as channel correlation is reduced (0.7 $\%$ for P_B=0.05, p_{corr}=1.0). However, the corresponding figures are reduced significantly as N is increased. Hence, under the given channel conditions, with typical values of N (N>2), we expect one-step prediction to approximate the throughput performance of ideal CSD-PCK closely.

B. Per-flow Delay-Loss Tradeoff

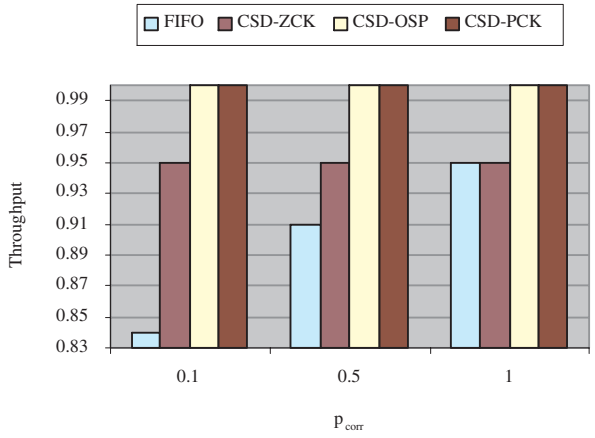
Consider the scheduling problem for delay-sensitive flows, and assume that each packet has to be transmitted within a specified delay bound when it becomes HOL, failing of which it will be dropped. To establish the capability of the FIFO and CSD schedulers to support such flows, we can compute the delay distribution, D_j(n), from Section IV and plot the function 1-D_j(n) as a function of n. Such a graph depicts the delay-loss tradeoff performance of the scheduler.

The delay-loss tradeoff obtained for N=2 is plotted in Fig. 9(a) and (b) for P_B = 0.01 and 0.05 respectively. The corresponding plots for N=3 and N=4 are shown in Fig. 10 and Fig. 11 respectively.

Each graph depicts the packet loss rate corresponding to a maximum delay bound obtained for each scheduler under a given channel condition. As an illustration, consider the packet loss rate when a maximum delay bound of 10 slots is specified for N=2, P_B=0.05. From Fig. 9(b), for p_{corr}=0.1, the loss rate for all the schedulers is between 10^{-2} and 10^{-3};

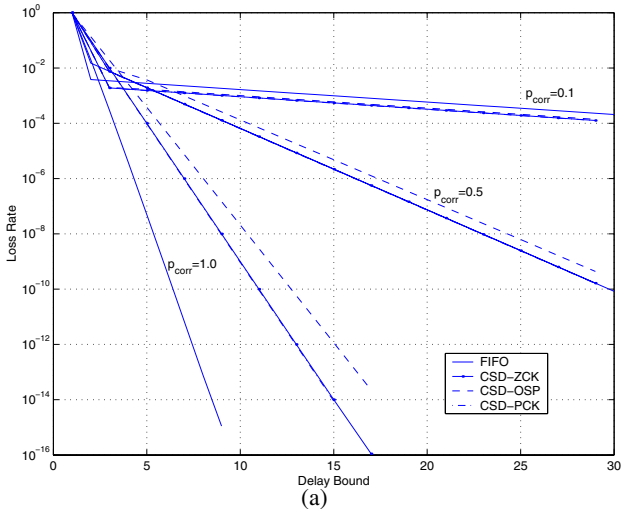


(a)

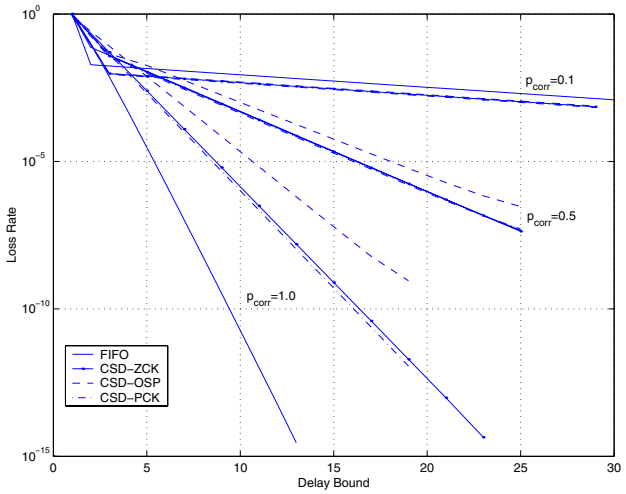


(b)

Fig. 8. Overall throughput for N=4 for (a) P_B=0.01 and (b) P_B=0.05



(a)



(b)

Fig. 9. Delay-loss tradeoff for N=2 for (a) P_B=0.01 and (b) P_B=0.05

for p_{corr}=1.0, the corresponding figures are about 10^{-5}, 10^{-7} and 10^{-11} respectively for the CSD-OSP, CSD-ZCK (CSD-PCK) and FIFO schedulers.

Several important observations are in place:

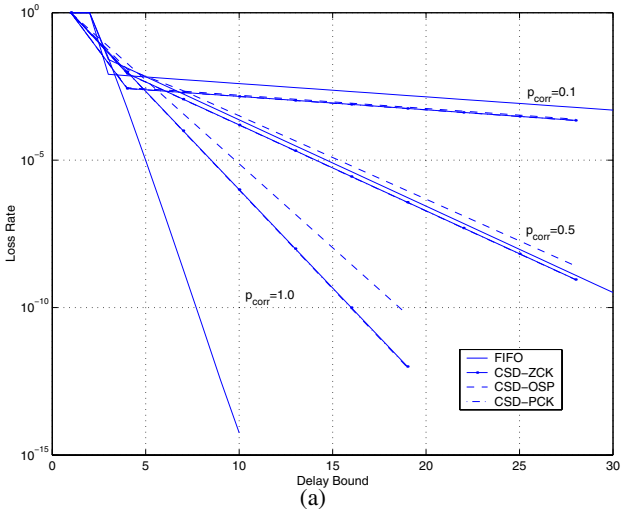
(a) For a given set of (N,P_B) values, CSD-PCK yields a marginal gain over CSD-ZCK in terms of delay-loss tradeoff. However, channel prediction (CSD-OSP) actually performs worse than CSD-ZCK and the difference becomes more significant as channel errors become less correlated due to the reduced accuracy of one-step prediction. However, the difference between CSD-PCK and CSD-OSP is reduced as N increases.

(b) For a given set of (N,P_B) values, FIFO scheduling performs worst when errors are highly correlated; however, its performance improves significantly as correlation is reduced such that it performs the best with zero channel correlation. Under such conditions, the deviation between FIFO scheduling and the other schedulers increases significantly as P_B and N is increased.

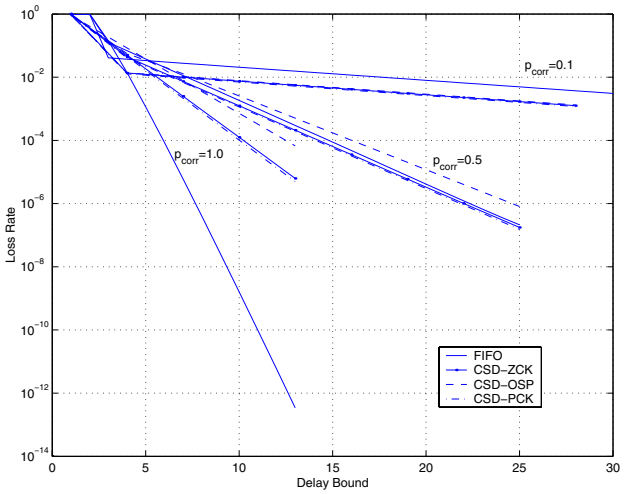
VI. Conclusions

In this paper, we studied the performance tradeoff at wireless links with various scheduling schemes. With FIFO scheduling, flows are aggregated into a single flow according to the rate weight of each flow just prior to the wireless link. Under Channel-State Dependent (or CSD) scheduling, the scheduler maintains a queue for each flow, and uses predicted channel information in addition to each flow's rate weight to make scheduling decisions. We consider three variants of the latter scheme: CSD-ZCK which does not make use of channel information, CSD-PCK which assumes perfect channel knowledge and CSD-OSP which uses one-step prediction to predict channel information.

FIFO scheduling is the simplest scheme since the scheduler needs only to maintain a single queue and does not require channel information in its scheduling decisions. Although it performs poorly in terms of throughput, it is ideally fair. In addition, this scheme achieves the best per-flow QoS performance in terms of loss-delay tradeoff when channel error

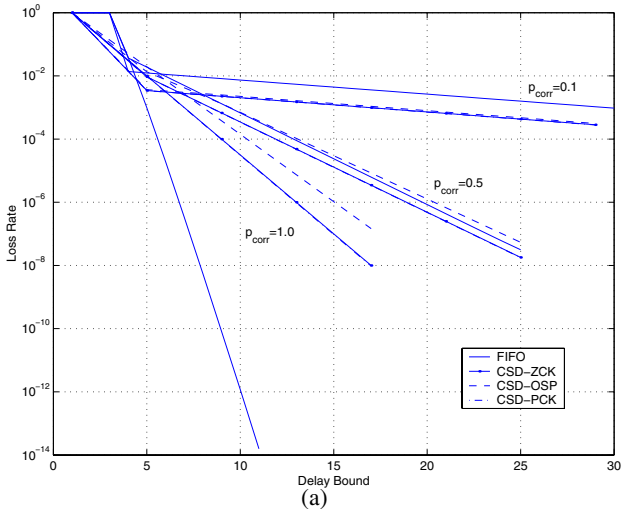


(a)

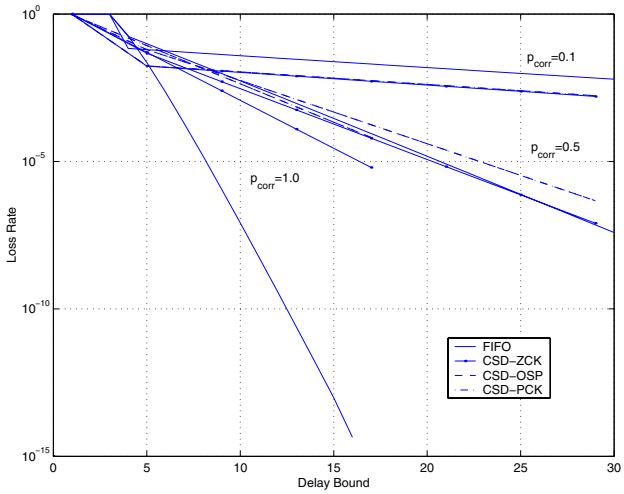


(b)

Fig. 10. Delay-loss tradeoff for N=3 for (a) P_B=0.01 and (b) P_B=0.05



(a)



(b)

Fig. 11. Delay-loss tradeoff for N=4 for (a) P_B=0.01 and (b) P_B=0.05

correlation is low.

CSD-ZCK retains the fairness properties of FIFO scheduling while improving its throughput performance when channel errors are correlated. This is achieved by maintaining one queue for each flow. However, the loss-delay tradeoff performance is degraded relative to FIFO scheduling under low channel error correlation.

CSD-OSP achieves almost the same throughput level as CSD-PCK (which achieves the maximum throughput) by using predicted channel information in its scheduling decision. However, this algorithm performs worst in terms of loss-delay tradeoff, especially when channel error correlation is low. However, the difference in the relative performance is reduced as the number of flows increases.

In future work, we intend to analyze the short-term performance of these scheduling schemes based on their Markov models. Good short-term performance is important for supporting real-time audio and video applications, which require low jitter. In addition, TCP performance degradation due to ACK compression has been observed due to poor short-term fairness.

REFERENCES

[1] Y. Guo and H. Chaskar, "Class-based quality of service oer air interfaces in 4G mobile networks," *IEEE Communications Magazine*, pp. 132–137, March 2002.

[2] P. Bhagwat, P. Bhattacharya, A. Krishna, and S. Tripathi, "Enhancing throughput over wireless lans using channel state dependent packet scheduling," *Proc. of the IEEE INFOCOM*, vol. 3, pp. 1133–1140, March 1996.

[3] S. Lu, V. Bharghavan, and R. Srikant, "Fair scheduling in wireless packet networks," *Proc. of the ACM SIGCOMM*, pp. 63–74, August 1997.

[4] T. Nandagopal, S. Lu, and V. Bharghavan, "A unified architecture for the design and evaluation of wireless fair queuing algorithms," *Proc. of the ACM MOBICOM*, pp. 132–142, October 1999.

[5] R. Rom and H. P. Tan, "Stochastic analysis of symmetric two-flow wireless-fair scheduling," CCIT Tech Report 371, Technion, Israel Institute of Technology, March 2002.