January 2007

Admission Control Framework to Provide Guaranteed Delay in Error-Prone Wireless Channel

Saswati Sarkar
University of Pennsylvania, swati@seas.upenn.edu

Prasanna Chaporkar
INRIA, France

Admission Control Framework to Provide Guaranteed Delay in Error-Prone Wireless Channel

Abstract
This paper considers a wireless system in which different sessions may use different channels with different transmission characteristics. A general framework for admission control and scheduling that provides stochastic delay and packet drop guarantees in this error-prone wireless system is proposed. By "general," the authors mean that the scheduling policies from a large class can be plugged in this framework and that admission control conditions can be obtained for different arrival processes. This enables the use of many scheduling policies that have not been considered so far for error-prone wireless systems. Using large deviation bounds and renewal theory, the authors prove that once a session *i* is admitted, irrespective of the scheduling policy and the channel errors experienced by other sessions, i obtains its desired quality of service. The admission control algorithm uses only individual channel statistics of sessions and not joint statistics, and the scheduling does not require any knowledge of instantaneous channel states.

Keywords
Admission control, delay guarantee, scheduling, wireless

Comments

Admission Control Framework to Provide Guaranteed Delay in Error-Prone Wireless Channel

Prasanna Chaporkar and Saswati Sarkar, *Member, IEEE*

***Abstract*—This paper considers a wireless system in which different sessions may use different channels with different transmission characteristics. A general framework for admission control and scheduling that provides stochastic delay and packet drop guarantees in this error-prone wireless system is proposed. By "general," the authors mean that the scheduling policies from a large class can be plugged in this framework and that admission control conditions can be obtained for different arrival processes. This enables the use of many scheduling policies that have not been considered so far for error-prone wireless systems. Using large deviation bounds and renewal theory, the authors prove that once a session i is admitted, irrespective of the scheduling policy and the channel errors experienced by other sessions, i obtains its desired quality of service. The admission control algorithm uses only individual channel statistics of sessions and not joint statistics, and the scheduling does not require any knowledge of instantaneous channel states.**

***Index Terms*—Admission control, delay guarantee, scheduling, wireless.**

I. Introduction

NEXT-GENERATION wireless packet networks will have to provide delay, delay jitter, and packet loss guarantees to heterogeneous real-time traffic. Emergency operations during disaster recovery and multimedia applications like on-demand video, distance learning, and teleconferencing cannot be supported without these guarantees. For these applications, every packet has a deadline, and packets that are not served before their deadlines must be dropped. Real-time applications can accommodate some packet loss without noticeable degradation in the quality of service (QoS), e.g., for voice, depending on the encoding and transmission schemes used, a 20% packet loss can be acceptable [1]. The tolerable loss is different for different applications. We focus on obtaining a resource allocation scheme that ensures the delivery of the required fraction of packets before the respective deadlines for each application in a wireless system.

Any framework that provides delay and packet loss guarantees needs 1) an "admission control mechanism" on the control path and 2) a "packet scheduling" on the data path. The admission control admits a session only if enough resources are available to limit its delay and packet loss, while the scheduling distributes the resources among admitted sessions in accordance with their delay and packet loss requirements. We now examine the challenges in designing admission control and scheduling schemes for wireless networks. First, admission control schemes in wireline networks cannot be easily extended for use in wireless systems. This is because location-dependent bursty channel errors in wireless systems lead to increased delays in delivering packets to destinations and consequently additional deadline violations and packet drops. This additional packet drop may cause unacceptable degradation in the quality of reception. Thus, the decision regarding whether to admit a new session must be different for wireline and wireless systems. For example, in Fig. 1, if there are no channel errors (wireline case), then the required delay guarantee can be provided to every packet of every session [Fig. 1(a)]. In the presence of channel errors (wireless case), the delay experienced by each packet is increased by one time unit [Fig. 1(b)]. If the packets that are delayed beyond their deadlines are dropped, then all the packets of session 3 will be dropped, and the remaining packets will be served [Fig. 1(c)]. Thus, sessions 3 suffers a 100% drop, whereas the drop rate for every session is 0 when there are no channel errors. Thus, unlike a wireline system, a wireless system cannot provide the desired QoS to the three sessions, and therefore, all these sessions should not be admitted.

Packet drops can be reduced in wireless systems by admitting fewer sessions and thereby reserving additional resources to compensate for the slots in which the channels have poor transmission quality. The resource reservations must just be sufficient to reduce the packet drop below the acceptable level as an increase in reservations leads to higher session blocking. The challenge is to "quantify" this excess reservation and use the quantification to design a channel-statistics-aware admission control algorithm.

A node may need to deliver packets of different sessions to different destinations (Fig. 2). Thus, different sessions may have different sequences of channel errors. The delay experienced by a session depends on the channel errors of all sessions. For example, in Fig. 1(b), session 3 experiences an additional delay of one slot although it never had a channel error. This has two undesirable consequences. First, the channel errors of a new session can affect the delay guarantees of the existing sessions. Thus, the joint channel state process of all the sessions has to be analyzed to determine whether the packet drops of the existing sessions remain below the required thresholds, even after a new session is admitted. Channel states for various

Manuscript received March 6, 2004; revised May 18, 2005, November 18, 2005, and February 13, 2006. This work was supported by the National Science Foundation under Grant NCR-0238340 and Grant CNS-0435306. The review of this paper was coordinated by Prof. S.-L. Kim.

P. Chaporkar is with INRIA, 78153 Paris, France (e-mail: Prasanna.Chaporkar@ens.fr).

S. Sarkar is with the Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, PA 19104 USA (e-mail: swati@seas.upenn.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVT.2006.883783

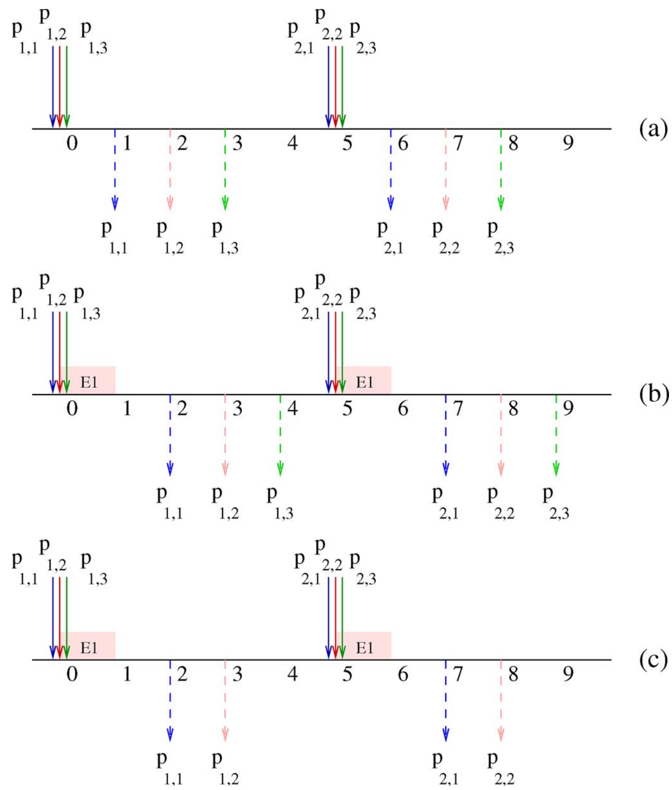

Fig. 1. Server that is serving packets for three sessions, namely, 1, 2, and 3. Each session has a leaky bucket-constrained arrival process with bucket depth (σ) as 1 and token replenishment rate (ρ) as 1/5. An arrival pattern that fits the above characterization is one that generates a unit length packet for each session at slots $0, 5, 10, \ldots$. A session can tolerate a maximum delay of three slots for each packet. Thus, all packets arriving at slot 0 have deadline 3. Let the EDF scheduling policy be used. Thus, the finish time of each packet equals its deadline. The packet with the least finish time is selected for transmission. Ties are always broken in favor of the session with the smallest index number. The arrows pointing toward and away from the axis represent packet arrivals and departures, respectively. Let $p_{k,i}$ denote the kth packet of session i. $E1$ indicates that session 1 has erroneous channel in the slot. The arrival process is the same for (a), (b), and (c), and the channel errors are the same for (b) and (c). (a) Service process for the wireline case (no channel errors). (b) Service process in erroneous channel if packets are not dropped even when their deadline expires. (c) Service process if a packet is dropped when its deadline expires.

sessions are correlated, and estimating this correlation is not feasible in practice. Second, excessive channel errors of one session may increase the packet drops of another session. For example, in Fig. 1(c), even when only session 1 has channel errors, all the packets of session 3 are dropped, while all the packets of session 1 are transmitted. Thus, if the admission control process uses an erroneous estimate of the channel statistics of a session, then other sessions may also have unacceptable packet drops.

We now outline our contribution. We consider a scenario where a single node transmits packets for different sessions using wireless channels to different destinations within its transmission range (Fig. 2). Sessions arrive at the node at different times and seek admission. We develop a framework that decides whether to admit a session given the delay, packet loss requirements, and channel statistics of the session and other existing sessions. The delay and packet loss requirements and channel statistics may be different for different sessions.

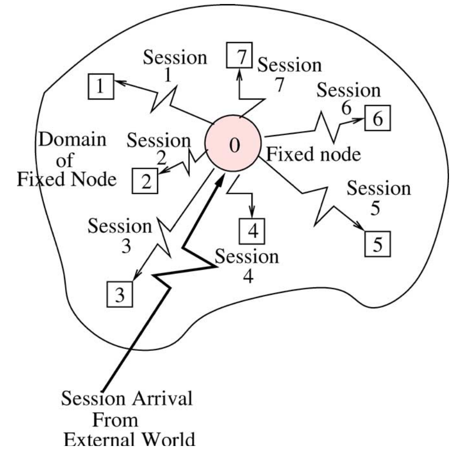

Fig. 2. Sessions arrive at a source node (node 0). Each session transfers data to one of the destination nodes (nodes $1, 2, \ldots, 7$), e.g., session 1 transfers data to node 1.

A session is admitted if there are enough resources to satisfy its delay and loss requirements without disrupting the guarantees for the existing sessions; otherwise, the session is blocked. Clearly, this decision depends on the particular scheduling policy used. We do not design a new scheduling policy, but design a method to obtain the admission control condition for any given scheduling policy. This enables the use of many scheduling policies that have not been considered yet for error-prone wireless systems. Consequently, any desired differentiation of service can be obtained among different sessions. Using large deviation bounds and renewal theory, we prove that once a session is admitted, irrespective of the scheduling policy, the session's expected packet drop rate is below its acceptable threshold, and the packets that are not dropped are delivered before their deadlines. The admission control algorithm uses only the individual channel statistics of sessions and not the joint statistics, and the scheduling does not require any knowledge of instantaneous channel states. The framework can, however, use the knowledge of the instantaneous channel states if these are known and further improve the performance. We also prove that a session's expected packet drop rate does not exceed its acceptable threshold even if the channel statistics of other sessions have been estimated incorrectly. Thus, the framework is robust.

This paper is organized as follows: In Section II, we review the relevant literature. In Section III, we describe our system model and notation. In Section IV, we present a general framework for admission control, which can provide delay and drop guarantees with any scheduling. In Section V, we show how to augment any given scheduling policy for use in the general framework. In Section VI, we present the performance guarantees for the framework. We conclude in Section VII. We present the proofs in the Appendix.

II. Literature Review

A prerequisite for developing a joint framework of admission control and scheduling to limit the delay and packet drop rate of sessions in wireless systems is to quantify the delay experienced by packets and the packet drop rates for different scheduling policies, arrival, and channel characteristics. Most of the prior work obtains delay guarantees for sessions that do not experience channel errors [2], [3]. The existing results that quantify delay in the presence of channel errors either 1) consider specific scheduling policies like FIFO [4] or scheduling policies that assign static priorities to sessions [5], or 2) consider a single channel [4]–[7], or 3) assume that before scheduling transmissions in a slot, the scheduler knows the state of the channel in the slot [4], [8], [9], and in some cases in all future slots [6], [7], or 4) upper bound the delay experienced by a packet after it reaches the head of line (HoL) position (HoL delay) [8], [9]. Note that in practice, before transmitting a packet, usually, the scheduler does not know the state of the channels in the current and future slots, and an upper bound for the HoL delay does not provide an upper bound for the overall delay experienced by a packet. Thus, one of our important contributions has been to provide a framework for quantifying the delay experienced by packets in the presence of 1) arbitrary scheduling policies, 2) multiple sessions, and 3) multiple channels whose transmission qualities vary with time and are not known at the scheduler before it schedules the transmissions. Thus, our framework can accommodate dynamic scheduling policies that alter the priorities of sessions depending on the deadlines and the service received so far.

Several authors presented scheduling policies that maximize throughput in the presence of channels with variable transmission quality [10], [11] and minimize the delay [11], [12] and the packet drop due to deadline expiry [13]. Depending on the traffic load and channel conditions, the packet drop and the delay of these optimal policies may not remain below an acceptable level. Admission control is necessary to limit the packet drop and the delay. The packet drop and the delay attained by these policies have either not been quantified [10], [11], [13] or quantified using computationally complex Markov decision processes (MDP) [12], but solving an MDP every time a session requests admission is not realistic. Thus, the results here cannot be easily extended to obtain a joint admission control and scheduling framework.

Admission control for minimizing the dropping of calls during handovers in cellular networks has been extensively studied [14]–[16]. Generally, these approaches "assume" that the amount of resources that must be reserved in a cell to provide the required QoS (e.g., signal-to-interference ratio [17], [18], average bandwidth, delay or packet loss [19]–[21]) is known and subsequently provide good heuristics to obtain desired balances between call dropping and call blocking [22]–[24]. In contrast, we concentrate on quantifying the resources required to deliver the desired QoS in the presence of channel errors. The schemes for cellular networks can use our analysis to determine the resources that need to be reserved.

Finally, a substantial amount of work has been done in the recent past to provide channel-state-aware scheduling in wireless networks (e.g., [8], [9], [25], [26]), but these do not seek to decrease the delay or the packet drop. The main goal in most of these has been to distribute available resources fairly among the sessions (e.g., [8], [9]).

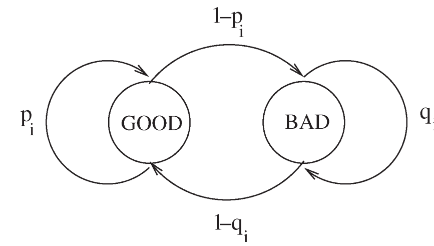

Fig. 3. Two-state MC that models bursty channel errors for session i.

III. System Model

We consider a dynamic scenario where sessions arrive at the source node and seek to transfer information to destinations via error-prone channels (Fig. 2). The destinations are within the transmission range of the source. The source decides whether to admit sessions and schedules the packets. Thus, the packet loss is only due to channel errors and not due to collisions or interference. Time is slotted. Each packet has unit length, arrives at the beginning of a slot, and can be served in the slot in which it arrives.

Each session i presents four tuples ($\vec{A}_i$, $\vec{C}_i$, D_i, and μ_i), where $\vec{A}_i$ and $\vec{C}_i$ are sets of parameters describing i's arrival process and channel statistics, respectively, and D_i and μ_i are the required delay and packet drop guarantees, respectively. Now, we explain each quantity.

We consider only deterministic traffic characterizations. A traffic characterization is deterministic if there exists a nondecreasing function $g(\tau)$ such that the maximum amount of data that the session can send in any duration τ is less than or equal to $g(\tau)$. For example, a (σ, ρ) leaky bucket-constrained traffic is a deterministic traffic, where $g(\tau) = \rho\tau + \sigma, \sigma$ is the bucket depth, and ρ is the token replenishment rate [27]. Session i's traffic is characterized by the function $g_i(\tau)$. Now, $\vec{A}_i$ is a set of parameters that characterize $g_i(\tau)$, e.g., $\vec{A}_i = \{\sigma_i, \rho_i\}$ for a leaky bucket-constrained session i.

We model the channel for each session by a two-state Markov chain (MC) (Fig. 3). If the channel state is "good" ("bad") for session i, then the probability of the successful transmission for session i is 1 (0). A packet is said to be "successfully transmitted" when it is transmitted in a slot in which its session's channel has a good state. The probability of transition of session i's channel from good (bad) state to bad (good) state is p_i (q_i). Thus, $\vec{C}_i = \{p_i, q_i\}$. This is the classical Gilbert–Elliot model [28], [29] and has been widely used to analyze the performance of wireless systems [4], [5], [13], [30]. Different sessions may have different channel statistics, depending on their destinations, power control, and channel coding schemes. The channel states of different sessions may be correlated. The source may not know the nature of such correlations.

Session i can tolerate delays of up to D_i slots. A packet P arriving at t should be successfully transmitted before $t + D_i$; otherwise, the packet is dropped. Thus, $t + D_i$ is the "deadline" for P. Let session i transmit n_i packets in all, $n_i < \infty$. The value of n_i may not be known when the session arrives. Let $\widehat{n}_i$ denote the number of session i packets dropped. Since the channel states are random, $\widehat{n}_i$ is a random variable. The expected packet drop rate of session $i, \mathbf{E}[\widehat{n}_i]/n_i$ must not exceed

μ_i for all possible sequences of packet arrivals of all sessions as long as the arrivals conform to the specified characterization $g_i(\tau)$, i.e.,

$$\frac{\mathbf{E}[\widehat{n}_i]}{n_i} \leq \mu_i. \tag{1}$$

Thus, the expectation is taken with respect to the channel state processes of the sessions.

Let session i seek admission. The admission control algorithm admits i if after i's admission (1) holds for i and other existing sessions and blocks i otherwise. Clearly, for any admitted session i, $\widehat{n}_i/n_i$ can be much greater than μ_i along some sample path; thus, (1) does not provide worst-case guarantees, but when each admitted session is in the system for a long duration, which is usually the case for many real-time applications, e.g., tele-conference, video-on-demand, etc., the fraction of packets dropped in most sample paths is close to $\mathbf{E}[\widehat{n}_i]/n_i$. We therefore consider the expected fraction of packets dropped as the QoS measure. Finally, note that if the sessions remain in the system for finite durations, no admission control and scheduling algorithm can ensure that at most μ_i fraction of packets is dropped for an admitted session i in every sample path if $\mu_i < 1$. This is because there exists a sample path of nonzero probability in which each admitted session drops all packets.

The admission control algorithm requires knowledge of only channel statistics that can be learned through effective estimation techniques [2]. We do not, however, assume knowledge of the instantaneous state of a channel at any scheduling instant. Thus, a session with "bad" channel may be served, and the transmitted packet can be received in error at the destination. If the deadline for this packet has not expired, then the packet may be retransmitted. We assume immediate feedback from each destination about the correctness of the received packet. Finally, the algorithms we present can also use the knowledge of instantaneous channel state and decrease the packet drop rate of the sessions if such information is available about some or all sessions.

IV. Admission Control Framework

We describe the admission control algorithm in this section. The admission control condition will depend on the precise scheduling scheme used. We first describe the class of scheduling strategies we consider. When a packet arrives, the scheduling policy assigns it a time stamp, the "finish time." The finish time of the packet is computed according to the specific scheduling policy, e.g., earliest deadline first (EDF) [31], first come first serve (FCFS), fair queueing policies like PGPS [32] (a.k.a. WFQ [33]), SCFQ [34], VirtualClock (VC) [35], WF^{2}Q [36], SFQ [37], a class of scheduling policies called rate proportional servers (RPS) [38], and the policies that promise certain service curves [39]. We discuss the computations of the finish times in Section V.

Now, we outline the scheduling. We present the details in Section V. In every slot, the scheduling policy determines which session would be served in the slot. This decision depends on the least finish time of packets in the sessions' queues and the number of packets of each session that had to be retransmitted. A session has higher priority if it has a packet with a low finish time and if it had fewer retransmissions. Knowledge of the channel states of the sessions, if available, is also incorporated in the decision process. Once the session is selected, the source transmits the packet in the session's queue that has the least finish time. Thus, the finish time is used as a priority indicator among the sessions and the packets of the same session. A large class of dynamic scheduling policies [31] may be implemented in wireless systems by appropriately selecting the algorithm for allocating the finish times.

Now, we explain the relation between admission control, a packet's finish time, and its deadline. Recall that the deadline of a packet from session i is the sum of the packet's arrival time and i's acceptable delay D_i. For a given scheduling policy, the primary goal of an admission control algorithm is to ensure that the expected fraction of packets dropped for any admitted session i is at most μ_i. This is sufficient to guarantee the required QoS. The admission control condition is different for different scheduling policies as the policies assign priority structure (by assigning finish times) in different fashion. Thus, the packet transmission sequence may differ for different policies. By successfully transmitting packets before their finish times, scheduling policies attain various objectives in addition to limiting the delay, e.g., PGPS guarantees short-term fairness [32], VC guarantees isolation [35], and SCED guarantees certain service curve [39]. If the scheduling strategy does not desire any property other than the acceptable delay, then it sets the finish time of each packet equal to the packet's deadline; this scheduling is referred to as EDF. We consider a class of scheduling policies that assign to each packet a finish time that is smaller than or equal to the packet's deadline. The class is large and contains all of the well-known policies mentioned above. The admission control strategy admits a session i only if in absence of channel errors each of its packets can be successfully transmitted before the packets' respective finish times, and in the presence of channel errors, its packets are successfully transmitted before their finish times with a probability of at least $1 - \mu_i$. Nevertheless, channel errors may delay a packet's successful transmission beyond its finish time. The session tries to transmit a packet until its deadline but drops a packet from the queue if it is not successfully transmitted before its deadline. This motivates the following definition.

Definition 1: Delay D_i is "guaranteeable" for session i if under every arrival stream that satisfies the session's traffic characterization, each packet of session i has a finish time less than or equal to the packet's arrival time plus D_i and is transmitted successfully before its finish time. Then, $\vec{D} = (D_1, \ldots, D_j)$ is guaranteeable if delay D_i is guaranteeable for each i.

We present the pseudocode for the admission control algorithm in Fig. 4. We next present the intuition behind the design.

Definition 2: A busy period is a sequence of contiguous slots such that the system has a packet to transmit in each slot in the sequence, and the system does not have a packet to transmit in the slots preceding and succeeding the sequence.

Let session i seek admission, and $\mathcal{C}$ be the set of existing sessions. If for a session j the expected fraction of packets

Procedure Admission_Control($i, \mathcal{C}$)
begin

/* Session i seeks admission */
/* $\mathcal{C}$ is the set of existing sessions */
/* $\pi_{G,j}$ ($\pi_{B,j}$) is the steady state probability of session j's channel being in *good* (*bad*) state */
STEP 1: /* Compute $(V_1, \ldots, V_{|\mathcal{C}|}, V_i)$ */
Define $\alpha_j = \frac{2q_j \pi_{B,j}}{1+q_j-p_j} + \pi_{G,j}$, $\beta_j = \frac{1+q_j-p_j}{q_j}$ and $\zeta_j = \frac{2q_j}{1+q_j}$ for very $j \in \mathcal{C} \cup \{i\}$.
(R1) $\widehat{Z} = \min_{t \geq 0} \left\{ \sum_{j \in \mathcal{C} \cup \{i\}} g_j[0,t] \leq \left(1 - \sum_{j \in \mathcal{C} \cup \{i\}} \frac{\log(\beta_j)}{\log\left(\frac{\beta_j}{\zeta_j}\right)} \right) t \right\}$.
For every $j \in \mathcal{C} \cup \{i\}$, obtain
(R2) $\delta_j = \frac{\log(\beta_j)}{\log\left(\frac{\beta_j}{\zeta_j}\right)} + \frac{\log\left(\frac{\alpha_j}{\min(\mu_j, \alpha_j)}\right)}{\widehat{Z} \log\left(\frac{\beta_j}{\zeta_j}\right)}$.
(R3) $Z = \min_{t \geq 0} \left\{ \sum_{j \in \mathcal{C} \cup \{i\}} g_j[0,t] \leq (1 - \sum_{j \in \mathcal{C} \cup \{i\}} \delta_i)t \right\}$.
$V_j = \delta_j Z$;
STEP 2: /* Check the guaranteeability of $\vec{D} - \sum_{j \in \mathcal{C} \cup \{i\}} V_j$ under perfect channel*/
$V = \sum_{j \in \mathcal{C} \cup \{i\}} V_j$;
if ($\vec{D} - V$ is guaranteeable under perfect channel) **then**
  admit session i;
  $\mathcal{C} = \mathcal{C} \cup \{i\}$;
else
  block session i;

end

Fig. 4. Pseudocode of a general admission control algorithm for an error-prone wireless channel.

dropped in every busy period is less than μ_j, then the overall fraction of packets dropped for session j is also upper bounded by μ_j. The admission control algorithm admits i only if after i is admitted, each session j's expected drop rate in each busy period is upper bounded by μ_j for every $j \in \mathcal{C} \cup \{i\}$. Let $\mathcal{B}$ be an arbitrary busy period. Let the "number of channel errors in $\mathcal{B}$" of a session j (E_j) be the number of slots in $\mathcal{B}$ in which j's channel has a bad state. A session needs to retransmit due to channel errors. Retransmissions for a session increase the delay for all sessions [Fig. 1(b)]. Intuitively, an upper bound on the delay experienced by a packet in the error-prone system is the sum of the delay under the perfect channel and $\sum_{j:j \in \mathcal{C} \cup \{i\}} E_j$. Here, "perfect channel" refers to a channel that is good with probability (w.p.) 1. Thus, if under perfect channel conditions, a packet can be successfully transmitted before its finish time, then under error-prone channels, it can be successfully transmitted before its finish time plus $\sum_{j:j \in \mathcal{C} \cup \{i\}} E_j$. If there exists a number V such that in each busy period $\sum_{j:j \in \mathcal{C} \cup \{i\}} E_j \leq V$, then the guaranteeability of delay $D_j - V$ for each j in $\mathcal{C} \cup \{i\}$ under perfect channel implies that under error-prone channels every packet of session j is transmitted successfully before its deadline. Thus, the packet drop is zero for each j in $\mathcal{C} \cup \{i\}$. Thus, session i can be admitted if the delay $D_j - V$ is guaranteeable for each session j in $\mathcal{C} \cup \{i\}$ under perfect channel, and blocked otherwise. The admission control algorithms in the wireline case can be used to check whether the above condition holds, but since the channel errors are random, the sum of the channel errors $\sum_{j:j \in \mathcal{C} \cup \{i\}} E_j$ in each busy period cannot be upper bounded by a constant V. Nevertheless, for a large enough V in each busy period, $\sum_{j:j \in \mathcal{C} \cup \{i\}} E_j$ can be less than or equal to V with a high probability.

Let $\mu_{\min} = \min\{\mu_j : j \in \mathcal{C} \cup \{i\}\}$. Consider the following two conditions.

Condition 1: Delay $D_j - V$ is guaranteeable for each session j in $\mathcal{C} \cup \{i\}$ under the perfect channel.

Condition 2: The sum of channel errors of admitted sessions in an arbitrary busy period $\mathcal{B}$ exceeds V w.p. at most $\mu_{\min}$.

If conditions 1 and 2 hold, it turns out that the expected fraction of packet drop for each session is less than or equal to $\mu_{\min}$. Thus, i can be admitted if there exists a V that satisfies conditions 1 and 2. The value of such V depends on the 1) channel characteristics of the sessions, 2) maximum length of a busy period, 3) $\mu_{\min}$, and 4) $(D_1, \ldots, D_{|\mathcal{C}|}, D_i)$. The challenge in designing an admission control algorithm then is to determine whether such V exists. Wireline admission control algorithms [40]–[44] can be used to verify whether a given V satisfies condition 1. Verification of condition 2 requires knowledge of an upper bound on the length of the busy periods in the presence of channel errors and the joint channel statistics of the states of all channels. A strategy to compute the upper bound is not known in the literature. Also, the source cannot compute the joint channel statistics from the individual channel characteristics unless it can estimate the statistical correlations among the channel states, which is difficult in practice.

Another problem with the above admission control scheme is that it admits session i only if the expected packet drop rate of i and each existing session is less than $\mu_{\min}$. Note, however, that $\mu_{\min}$ may be less than the expected packet drop rate requested by several sessions. Thus, the admission control condition has an unnecessarily high session blocking.

The above problems can be addressed if the system satisfies the following additional properties for some vector $(V_1, \ldots, V_{|\mathcal{C}|}, V_i)$ such that $V = \sum_{j \in \mathcal{C} \cup \{i\}} V_j$.

Condition 2′: The probability that the number of channel errors of j in $\mathcal{B}$ exceeds V_j is less than or equal to μ_j.

Condition 3 (Decoupling condition): If the number of channel errors of j in every busy period is less than or equal to V_j, then each packet from session j that arrived in $\mathcal{B}$ is successfully transmitted before its finish time plus V.

Let the system satisfy conditions 1, 2′, and 3 for some $(V_1, \ldots, V_{|\mathcal{C}|}, V_i), V = \sum_{j \in \mathcal{C} \cup \{i\}} V_j$. It then turns out that the expected fraction of packet drop for each session j is less than or equal to the desired quantity μ_j for each $j \in \mathcal{C} \cup \{i\}$. Thus, i can be admitted if conditions 1, 2′, and 3 hold. We now discuss how to verify whether conditions 2′ and 3 hold.

Condition 3 decouples the performance of different sessions as it ensures that once a session is admitted, its packet drop guarantees do not depend on the number of channel errors of other sessions. We later discuss why decoupling is a good feature of any system. We now demonstrate that some common scheduling policies do not provide this decoupling.

Example 1: Consider the wireless system shown in Fig. 1(c). For each $i \in \{1, 2\}$, in any slot, session i's channel has a bad state w.p. p, independent of its channel state in other slots and the channel states of other sessions. Session 3 has a perfect channel. Let $\mu_1 = \mu_2 = 1 - (1-p)^3, \mu_3 = 0$. Session 3, however, drops a packet every time session 1 has a bad channel in the slot in which 1 is served [Fig. 1(c)]. This has a nonzero probability. Thus, session 3 cannot be guaranteed a zero packet drop rate in the presence of sessions 1 and 2 and, therefore, should not be admitted. Since session 3 can drop packets even when it does not have any channel error, a V_3 that satisfies Condition 3 does not exist.

We now show how to modify the scheduling in Fig. 1(c) so as to satisfy condition 3, which would in turn provide the required packet drop guarantees. Let the scheduler transmit packets with the minimum deadline as before, but now it drops a packet if the first transmission is not successful. Thus, each busy period has length 3. Let $V = 0$. Thus, $V_i = 0$, for all i. Thus, $\vec{D} - V$ is guaranteeable under perfect channel and hence condition 1 holds. The probability that the number of channel errors of session i exceeds 0 in a busy period is at most μ_i for each $i \in \{1, 2, 3\}$. Thus, condition $2'$ holds. Condition 3 holds as a packet is dropped only if its session has a slot in which its channel is erroneous. The source admits sessions 1, 2, and 3 simultaneously. Sessions 1 and 2 have drop rates of $p \leq \mu_1 = \mu_2$, and session 3 has a drop rate of 0. ■

In Section V, we provide a general framework to augment any scheduling to decouple the performance of different sessions. Specifically, we will ensure that condition 3 holds if conditions 1 and $2'$ hold. Thus, the admission control algorithm in Fig. 5 admits a session i only if conditions 1 and $2'$ hold in the presence of i.

We now discuss the two major steps of the admission control algorithm (Fig. 4). In Step 1, the algorithm first computes a quantity Z that upper bounds the length of each busy period with high probability. Then, using the knowledge of Z, the algorithm computes $(V_1, \ldots, V_{|\mathcal{C}|}, V_i)$ such that condition $2'$ holds. In Step 2, the algorithm checks condition 1 with $V = \sum_{j \in \mathcal{C} \cup \{i\}} V_j$. Note that whether $\vec{D} - V$ is guaranteeable under perfect channels can be checked using Network Calculus [41], [42] for several policies, e.g., EDF [31], FCFS, PGPS [32], SCFQ [34], VC [35], WF^{2}Q [36], SFQ [37], RPS policies [38], and the policies that promise certain service curve [39]. The new session is admitted if both conditions 1 and $2'$ hold; otherwise, it is blocked.

Now, we discuss how in Step 1 we obtain $(V_1, \ldots, V_{|\mathcal{C}|}, V_i)$ that satisfy condition $2'$. Suppose we obtain a function $\phi_j(T)$ (channel error rate) that satisfies the property that the number of channel errors for session j in duration T exceeds $T\phi_j(T)$ with probability at most μ_j. Let Z_1 denote an upper bound on the length of the busy period. Then, to satisfy condition $2'$, it suffices to choose $V_j = Z_1 \phi_j(Z_1)$. Such $\phi_j(T)$ can be obtained using Chernoff bounds for any T. A problem, however, is that Z_1 is a random variable, and $\phi_j(Z_1)$ depends on Z_1. For example, a higher channel error rate corresponds to a longer busy period. Hence, we first obtain a lower bound $\widehat{Z}$ on Z_1, which depends only on the channel statistics [relation (R1) in Fig. 4]. Then, we set $\delta_j = \phi_j(\widehat{Z})$ [relation (R2) in Fig. 5]. Finally, we obtain an upper bound Z on the length of any busy period when the decoupling condition (condition 3) holds with $V_j = \delta_j Z$ [relation (R3) in Fig. 4]. Since $\phi_j(Z) \leq \phi_j(\widehat{Z})$ and $Z \geq \widehat{Z}$, $V_j \geq Z\phi_j(Z)$. Thus, the probability that the number of channel errors for session j exceeds V_j is at most μ_j. Therefore, condition $2'$ holds for the $(V_1, \ldots, V_{|\mathcal{C}|}, V_i)$ obtained in Step 1.

```
Procedure Scheduling_Interface()
begin
  When a new slot t begins
  Assign finish times to packets that arrive in the slot;
  C_{primary-busy} = {j : prim_queue(j) is not empty};
  C'_{comp-busy} = {j : comp_queue(j) is not empty};
  ft_j = the finish time of HoL packet in prim_queue(j) ∀j ∈ C_{primary-busy};
  ft'_j = the finish time of HoL packet in comp_queue(j) ∀j ∈ C'_{comp-busy};
  If a new primary busy period begins at t, Comp_counter(j) = 0 for every session
  j;
  if (C_{primary-busy} ≠ φ) then
    /* If some session has a packet in primary queue */
    i = arg min_{j∈C_{primary-busy}} {ft_j};
    if (Transmit_comp_packet(i) = FALSE) then
      /* session i does not have packet with valid deadline in compensation queue */
      transmit HoL packet of prim_queue(i);
    if (the packet is received without error) then
      /* i had good channel */
      delete the transmitted packet from the corresponding queue;
      if (a packet is transmitted from comp_queue(i)) then
        Transfer HoL packet of prim_queue(i) to comp_queue(i);
    else
      /* i had a bad channel */
      Comp_counter(i) = Comp_counter(i) + 1;
      if (Comp_counter(i) > V_i) then
        /* i has already taken allotted compensation */
        put HoL packet of prim_queue(i) into comp_queue(i);
  else
    flag = FALSE;
    while ((C'_{comp-busy} ≠ φ) AND (flag = FALSE)) do
      i = arg min_{j∈C'_{comp-busy}} {ft'(j)};
      flag = Transmit_comp_packet(i);
end

Procedure Transmit_comp_packet(i)
begin
  transmit_flag = FALSE;
  while (comp_queue(i) is not empty) do
    if (finish time of HoL packet in comp_queue(i) ≤ curr_time) then
      drop the packet;
    else
      transmit HoL packet of comp_queue(i);
      transmit_flag = TRUE;
      exit while loop;
  return(transmit_flag)
end
```

Fig. 5. Pseudocode for a general scheduling framework.

V. Scheduling Framework

Several scheduling policies are known for wireline and wireless networks. We now present a framework that allows the system to use any desired scheduling policy and still attain the desired delay and packet drop guarantees for each admitted session. We first assume that the scheduler does not know the channel state of any session in a slot before deciding which session to serve in the slot. We later discuss how to utilize knowledge about instantaneous states of channels if such information is known.

The framework we propose decouples the performance of different sessions. More precisely, the framework ensures that irrespective of the arrival traffic and channel conditions of other sessions, an admitted session i does not experience packet drop in any busy period in which it has at most V_i channel errors. The parameter V_i is determined by the admission control algorithm (Fig. 4) during i's admission and depends on the maximum delay D_i and the maximum packet drop rate μ_i specified by i. This decoupling ensures that the QoS guarantees of different sessions are satisfied. We first present a definition.

Definition 3: The compensation slot of a session is a slot in which its packet is transmitted, and its channel has a "bad" state.

A packet may be transmitted when the session has a bad channel as channel states are not known before transmission. If a session's packet is transmitted when its channel has a bad state, then the packet must be retransmitted or dropped. Retransmission causes additional delays and subsequently deadline expiry and packet drop. Fig. 1(c) illustrates that the packet drop of a session depends on the retransmissions of other sessions. We decouple the performance of different sessions by limiting the number of retransmissions of each session. Let a session i's packet be received in error. If the number of compensation slots of a session i in a busy period is less than V_i, then the packet is retransmitted; otherwise, the packet may be dropped depending on the system load. Thus, V_i can be looked upon as the retransmissions guaranteed to session i to compensate for its channel errors. This strategy ensures that excessive channel errors of a session do not lead to a large number of retransmissions and hence additional delays and packet drops for other sessions.

In Fig. 5, we present the pseudocode for the scheduling framework. We explain the operations here. Each session i maintains two queues, namely 1) a primary queue $(\text{prim\_queue}(i))$ and 2) a compensation queue $(\text{comp\_queue}(i))$. New packets of a session join at the end of its primary queue. The sessions that have packets in their primary queue are called "primary busy sessions." Similarly, the sessions that have packets in their compensation queue are called "comp-busy sessions." A "primary busy period" is a sequence of contiguous slots such that the system has at least one primary busy session in each slot in the sequence and no primary busy session in the slots preceding and succeeding the sequence. Each session i maintains a "compensation counter" $(\text{Comp\_counter}(i))$, which is the number of compensation slots of the session in the current primary busy period. After the compensation counter exceeds V_i, the packets that need retransmission are transferred to the session's compensation queue. When a primary busy period ends, packets from the compensation queues are transmitted in increasing order of their respective finish times, unless their deadlines have expired. We now discuss the algorithm in detail.

The scheduler assigns a finish time to each new packet. Let $p_{k,j}$ denote the kth packet of the jth session. Also, let $a_{k,j}$ and $f_{k,j}$ denote the arrival time and the finish time, respectively, for $p_{k,j}$. For example, $f_{k,j} = a_{k,j} + D_j$ under EDF, and $f_{k,j} = \max\{a_{k,j}, f_{k,j}\} + 1/r_j$ under VC, where r_j is j's weight. We assume that $f_{k,j} \leq f_{k+1,j}$ for all k, j, but the ordering among the finish times of packets of different sessions need not be the same as that between their arrival times.

Clearly, the HoL packet in a session's primary queue has the minimum finish time among all packets in the session's primary queue. The finish time $ft(j)$ for a primary busy session j is the finish time of the HoL packet in j's primary queue. The system transmits the HoL packet in the primary queue of the primary busy session with the minimum finish time. Let this session be i. If the transmission is successful, the packet is removed from the system; otherwise, the scheduler increments $\text{Comp\_counter}(i)$. If $\text{Comp\_counter}(i)$ is less than or equal to V_i, the packet is retained at the HoL position in the primary queue; otherwise, the packet is transferred to the compensation queue.

When the system does not have a primary busy session, it transmits a packet that has the minimum finish time among all packets in the compensation queues. In every slot, the system drops from each session's compensation queue the packets whose deadlines have expired.

The algorithm we have described does not deliver packets in order. For example, let a session j packet $p_{k,j}$ be transferred to j's compensation queue at slot t. Let $t+1$ j be selected for service, let $p_{k+1,j}$ be in its primary queue, and let j's channel have a "good" state. Then, $p_{k+1,j}$, which arrived after $p_{k,j}$, is delivered to the destination at $t+1$. Now, $p_{k,j}$ will be delivered to the destination after $t+1$, when j's primary queue becomes empty. The following simple modification ensures that the packets are transmitted in order. Whenever session j is selected, a) the HoL packet in j's compensation queue is transmitted if j's compensation queue is nonempty, and b) the HoL packet in j's primary queue is transmitted otherwise. If the transmission is successful in (a), then the HoL packet from j's primary queue is transferred to j's compensation queue, and the transmitted packet is removed from j's compensation queue. If the transmission is not successful in case (a), then $\text{Comp\_counter}(j)$ is incremented, and the transmitted packet is retained at the HoL position of the compensation queue. If $\text{Comp\_counter}(j) > V_j$, the HoL packet from j's primary queue is transferred to j's compensation queue. The rest remains the same.

We show that before a packet's deadline expires, it is either successfully transmitted or transferred to its compensation queue. Since in every slot the system drops from the compensation queue packets whose deadlines have expired, packets are never transmitted after their deadlines. We show that μ_j upper bounds the expected fraction of packets dropped for session j.

Observe that although in a primary busy period we may transmit a packet from compensation queue, we select the session that receives service on the basis of finish times of packets in the session's primary queue. To summarize, in a primary busy period, a session j is scheduled if it has the smallest finish time among all the primary busy sessions, but the first packet from the primary or compensation queue for session j, whose deadline has not expired, is transmitted. When a system does not have a primary busy session, it transmits packets in increasing order of finish times from the compensation queues of all sessions. This prevents packets that represent additional retransmission from contending for channel access. Packets from compensation queues contend for channel access when the primary queues are empty; then, unused resources can be used for additional retransmissions.

Finally, we describe how the framework can be modified to improve performance when the source knows the channel states of all sessions at the beginning of each slot. Now, in a slot in a primary busy period, a primary busy session that has a good channel and has the minimum finish time among all the primary busy sessions with good channels is scheduled. A compensation slot for a session is one in which it is primary busy, it has a bad channel, and it has the minimum finish time among all primary busy sessions. The compensation counter for a session is incremented in each of its compensation slots. When the system has no primary busy session, packets are transmitted in increasing order of their finish times from the sessions with good channels. The rest remains the same.

VI. Performance Evaluation

We present the analytical guarantees provided by the joint admission control and the scheduling framework. All results in this section hold irrespective of whether instantaneous channel states are known before scheduling. In the Appendix, we present proofs for the case when the scheduler does not know the instantaneous channel states. The proofs for the case in which the scheduler knows the channel states for the sessions in every slot follow using similar arguments.

Lemma 1: If a packet is transmitted successfully at a slot t, then its deadline is greater than or equal to t.

Lemma 1 ensures that a packet does not reach its destination after its deadline. We next show that the scheduling interface decouples the packet drop for various sessions, even in the presence of channel errors.

Lemma 2: If in every primary busy period the number of channel errors for session i is at most V_i, then there is no packet drop for session i. Further, every session i packet is successfully transmitted before the packet's finish times plus V.

Lemma 3: If the probability that the number of slots in which session i's channel has a bad state in a primary busy period exceeds V_i is at most μ_i, then the expected fraction of packets dropped for session i is at most μ_i.

Lemma 2 shows that a session i does not have any packet drop if i's channel has at most V_i erroneous slots in each primary busy period. Now, the admission control algorithm admits session i if the number of channel errors for i in each primary busy period is less than V_i with probability μ_i. From Lemma 3, this ensures that session i's expected packet drop rate is μ_i. Let in each primary busy period the number of channel errors of an admitted session j be significantly larger than V_j. This may happen if j's channel parameters are not estimated correctly when j seeks admission. Lemma 3 ensures that in this case, only j's guarantees are violated. Several existing scheduling policies do not have this desirable feature. This is because existing scheduling policies serve the session that has the packet with the minimum finish time in the system. Thus, if a session has several channel errors, then its packets remain in the system for long durations but are nevertheless repeatedly selected for service because of low finish times. This delays the transmission of packets of other sessions as well. Our framework serves the session with the minimum finish time, where the finish time of a session is the finish time of the HoL packet in the session's primary queue. When a session has several channel errors, its packets with low finish times are transferred to the compensation queue and are not considered while deciding which session to serve. The following experiment illustrates this feature.

Consider a system with two sessions. Let the estimated channel parameters for the sessions be $p_1 = p_2 = 0.999$ and $q_1 = q_2 = 0.9$. These channel parameters correspond to a Raleigh fading channel that has good channel of 99% time and mean fade duration of ten slots [13]. Let the actual channel parameters for session 1 equal the estimated values, but let the actual channel parameters for session 2 be different from the estimated values. We consider leaky bucket-constrained sources. We assume that $\sigma_i = 10$, $\rho_i = 0.3$, $\mu_i = 10^{-3}$, and $D_i = 30$ for $i \in \{1, 2\}$. Let the finish time of each packet equal its deadline (EDF). We study the fraction of packets dropped for the sessions under our framework (Table I) (augmented EDF) and when the packet with the minimum deadline is served (Table II) (plain EDF). We run simulations for 10^6 slots. The column $1 - \hat{q}_2$ refers to the actual value of channel parameter for session 2. Under plain EDF, packet drops for both sessions increase with the increase in the difference between session 2's actual value of the channel parameter and the estimated value (Table II), but under our framework, the packet drop increases only for session 2 and is almost constant for session 1 (Table I). Thus, unlike under plain EDF, in our framework, the required packet drop guarantee is always provided to session 1, irrespective of the channel conditions for session 2.

TABLE I
PACKET DROP WITH EDF SCHEDULING UNDER THE PROPOSED FRAMEWORK

$1 - \hat{q}_2$	Packet Drop for Session 1 (%)	Packet Drop for Session 2 (%)
0.1	0.01647	0.02987
0.075	0.01560	0.09093
0.050	0.01773	0.29313
0.025	0.01933	1.24013

TABLE II
PACKET DROP WITH PLAIN EDF SCHEDULING

$1 - \hat{q}_2$	Packet Drop for Session 1 (%)	Packet Drop for Session 2 (%)
0.1	0.02713	0.03467
0.075	0.05440	0.06787
0.050	0.15160	0.17940
0.025	0.60987	0.66433

Finally, in the following theorem, we show that the joint admission control and scheduling framework provides the required QoS guarantees.

Theorem 1: In the joint admission control and scheduling framework (Figs. 4 and 5), the expected fraction of packets dropped for every admitted session i is at most μ_i.

Thus, the reservations provided by our framework are sufficient to guarantee the desired delay and loss guarantees. Using an example, we now demonstrate that the reservations are also necessary. Specifically, the following example shows that if a session i is admitted when there does not exist a $\vec{V}$ that satisfies conditions 1 and 2′, then the expected fraction of packets dropped for i or an existing session j may exceed μ_i or μ_j, respectively. Thus, the guarantees provided by the framework are stochastically tight.

Example 2: Consider the arrival pattern shown in Fig. 1(c). Let sessions 1 and 2 be in the system when session 3 seeks admission. The delay requirement for each session is 3. Thus, the busy period length is at most three, irrespective of whether session 3 is admitted. Sessions 1 and 2 have independent identically distributed channel errors. The probability that session 1 has a "bad" channel in any slot is p, i.e., $1/2 < p < 1$. Let $\mu_1 = 3p^2(1-p) + (1-p)^3$ and $\mu_2 = (1-p)^3$. Session 3 has a perfect channel, and $\mu_3 = p/2$. Before session 3 seeks admission, $V_1 = 1$ and $V_2 = 0$ satisfy conditions 1 and $2'$. Session 3 is not admitted as there does not exist V_1, V_2, and V_3 that can satisfy conditions 1 and $2'$.

Now, we show that if session 3 is admitted, then the drop guarantees for at least one session are violated under any assignment of V_i. Clearly, either (a) $V_1 = 0$ or (b) $V_1 > 0$. Now, (a) violates $2'$ for session 1. Here, session 1's packet will be dropped if one has a channel error at slot 0. Thus, the expected fraction of packets dropped for session 1 is p. Since $p > \mu_1$, session 1's drop guarantee is violated. Next, (b) violates condition 1. Here, session 3's packet will be dropped if session 1 has a channel error at slot 0. Thus, the expected fraction of packets dropped for session 3 is p. Since $p > \mu_3$, session 3's packet drop guarantee is violated.

We have demonstrated that the reservations provided by our framework are necessary in statistical worst-case scenarios. The specific worst cases we considered are the following.

1) The framework provides reservations based on the assumption that the channel errors for various sessions occur in distinct slots.
2) The framework assumes that if any session has a bad channel, then the server remains idle.

If, however, these worst cases do not occur, then our framework is conservative in the sense that its session blocking may be higher than the minimum required for providing the required statistical guarantees to the admitted sessions. Nevertheless, considering the statistical worst cases is imperative from practical considerations. We now explain why this is the case. The channel errors for different sessions may not always occur in different slots, but this fact can only be exploited (i.e., (1) can be relaxed) if the correlations between the channel error processes of different sessions are known precisely. The correlations between the error processes can be determined only when all statistical moments of the joint error processes are known. Estimating all moments of the joint error processes is difficult, if not impossible, in practice. In practice, a node may only be able to estimate the first-order statistics (expectations) of the individual error processes. Thus, a node needs to provide statistical guarantees using only the first-order statistics (expected error rates) and hence needs to consider the worst possible correlations of the error processes of different sessions in providing such guarantees. This is exactly what our framework does. Also, to relax assumption (2), knowledge of the correlation between arrival and channel error processes is required. There exists arrival and channel error patterns such that (2) is true for any scheduling policy (Fig. 1 shows an example for EDF scheduling policy). Again, estimating the correlation between the arrival and channel error processes is difficult in practice, and hence, the worst possible correlations between these must be assumed for providing statistical guarantees.

Furthermore, we expect that the admission control schemes will be executed by service providers. Oftentimes, service providers have service contracts with the customers that obliges the providers to provide the mutually agreed service guarantees. In these cases, even minor violations of the mutually agreed guarantees require the provider to refund the entire service fee. The service providers often therefore prefer to admit the sessions only when they obtain the required service even under the worst possible statistical combinations, which in turn requires them to use conservative schemes.[1] Thus, we expect them to find our scheme useful.

In the following lemma, we prove one more useful property of our framework.

Lemma 4: Packets from a session reach the session's destination in increasing order of their arrivals.

All the analytical guarantees hold for more general models (e.g., with more than two states [46]). In these Markovian models, the state u corresponds to a loss probability of ℓ_u, where $0 \leq \ell_u \leq 1$. The only distinction in the analysis is that the Chernoff bounds can no longer be used to upper bound the expected number of channel errors in a busy period. The upper bound can, however, be obtained using large deviation theory.

Finally, the computational complexity of our framework depends on the scheduling algorithm and the arrival processes. Depending on the scheduling algorithm and the arrival process, the scheme may require per-flow states. For example, when the scheduling is either EDF or PGPS, and the arrivals are either leaky bucket-constrained or constant bit rate, the computational complexity of the framework is $O(N^2)$, where N is the number of sessions active at a node. However, N is typically small and is not likely to increase significantly due to the limitations in available bandwidth. Hence, maintaining per flow states is not a serious deterrent, and the framework is computationally feasible. If, however, N does become large in the future, then existing or new scheduling algorithms that do not need per flow states will be implemented. Now, the admission control algorithms may still need per-flow states, but these states will be used only when a session arrives or departs and not during the transmission of each packet. Thus, the computational complexity of our framework remains low even for large N.

VII. Conclusion

Summarizing, we have proposed a joint framework for admission control and scheduling that delivers desired delay and packet drop guarantees to sessions traversing error-prone wireless channels. Such a framework can only be developed if the amount of resources that need to be reserved for each session can be quantified. Such quantification is nontrivial, e.g., even if only one session is admitted, it is not necessary that its delay and loss guarantees will be satisfied particularly when the wireless channel has poor transmission quality and the delay and loss guarantees are stringent. Thus, determining how much reservation will suffice is critical, unless the system

[1] Specifically, a customer i provides a delay requirement of D_i and a loss requirement of μ_i. It does not accept a packet if it is delivered D_i units after it is generated and considers such a packet to be lost. A customer demands a refund if more than μ_i of a fraction of its packets is lost. Since the channel error process is stochastic, more than μ_i of a fraction of i's packets may be lost for any given admission control and scheduling scheme. Thus, the provider may indeed have to refund service fees for some customers, irrespective of what it does, but we show that by considering the worst possible statistical combinations, the provider forfeits fees only with negligible probability [45].

```
Procedure Scheduling_for_E()
begin
  When a new slot t begins
  Assign finish times to packets that arrive in the slot;
  C_busy = {j : session j has a packet};
  ft_j = finish time of HoL packet of session j ∀j ∈ C_busy;
  If a new busy period begins at t, Comp_counter(j) = 0 for every session j;
  if (C_busy ≠ φ) then
    /* If some session has a packet */
    i = arg min_{j∈C_busy} {ft_j};
    transmit HoL packet of session i;
    if (the packet is received without error) then
      /* i had good channel */
      delete the transmitted packet from the system;
    else
      /* i had a bad channel */
      Comp_counter(i) = Comp_counter(i) + 1;
      if (Comp_counter(i) > V_i) then
        /* i has already taken allotted compensation */
        drop HoL packet of session i;
end
```

Fig. 6. Pseudocode for scheduling in E.

chooses the trivial policy of rejecting all sessions (i.e., 100% call blocking ratio). Using large deviation bounds and renewal theory, we provide a systematic approach for such quantification. The framework can accommodate scheduling policies from a large class and obtain admission control algorithms for any given scheduling in this class and different arrival processes. We prove that once a session is admitted, irrespective of the scheduling policy, the session's expected packet drop rate is below its acceptable threshold, and the packets that are not dropped are delivered before their deadlines. The proposed framework is robust since an inaccurate estimation of channel statistics does not affect the performance of other existing sessions in the system. Furthermore, admission control only uses individual channel statistics of the sessions and not joint channel statistics, and the scheduling need not know the instantaneous channel state. The framework can, however, use the knowledge of the instantaneous channel states, if these are known, and further improve the performance.

Appendix

This Appendix is organized as follows: First, we introduce notations. Then, we state the supporting lemmas (Lemmas 5–8) and use these to prove the performance guarantees (Lemmas 1–4 and Theorem 1). Finally, we prove the supporting lemmas.

Notation

In this section, we discuss important notations and terminologies used in the proofs. We denote the joint admission control and scheduling scheme proposed in Sections IV and V as EC. We consider two fictitious systems, namely 1) PC and 2) E. The only difference between EC and PC is that PC does not have channel errors. The only difference between EC and E is that E does not have compensation queues, and hence, each session i's packet is dropped whenever the number of compensation slots for i exceeds V_i in a busy period. For convenience, in Fig. 6, we give the pseudocode for the scheduling in E.

TABLE III
Notations Used in the Appendix

Notation	Definition
$\mathcal{C} = \{1, \dots, n\}$	Set of sessions present in the system.
$p_{k,i}$	The k^{th} packet of session i.
$a_{k,i}$	The arrival time of $p_{k,i}$.
$f_{k,i}$	The finish time of $p_{k,i}$.
$d_{k,i}$	The departure time of $p_{k,i}$. A packet departs when it is transmitted successfully or dropped.
$\gamma[t, t+\tau]$	The number of packets that arrived in $[t, t+\tau]$.
$w[t, t+\tau]$	The number of packets that arrived in $[t, t+\tau]$ with finish time less than or equal to $t+\tau$.
$S[t, t+\tau]$	The number of packets transmitted successfully in $[t, t+\tau]$.
$\widehat{S}[t, t+\tau]$	The number of packets that depart in $[t, t+\tau]$.

We first upper bound the expected delay and packet drop rates in PC and E. Then, we relate the performance in EC with those in PC and E. Using these relations, we prove Lemmas 1–4 and Theorem 1.

EC, PC, and E have the following features.

A1) All these systems admit the same set of sessions and have the same arrivals.

A2) The finish times of packets are the same in PC and EC. The finish time of a packet in E equals that in PC plus V. Thus, for every k and session i

$$f_{k,i}^{E} = f_{k,i}^{PC} + V = f_{k,i}^{EC} + V. \tag{2}$$

A3) The channel state for every session i is the same at every slot t in EC and E. Under PC, the channel state is always "good."

A4) In PC and E, a packet with the smallest finish time is transmitted at every slot. If multiple packets have the same finish time, then the packet from the session with the smallest index is transmitted.

The basic notations used here are defined in Table III. These symbols will be augmented to denote various dependencies. A superscript EC, PC, or E will indicate quantities corresponding to the systems EC, PC and E, respectively. Further, the subscript i will indicate quantities corresponding to session i. The absence of subscript will indicate that the quantities correspond to the whole system. For example, $S_i^{PC}[t, t+\tau]$ is the number of successfully transmitted packets of session i in PC in the interval $[t, t+\tau]$, and $S^{PC}[t, t+\tau] = \sum_{I\in\mathcal{C}} S_i^{PC}[t, t+\tau]$.

Supporting Lemmas Used in the Proofs of Results in Section VI

We state the supporting lemmas (Lemmas 5–8) that we used to prove Lemmas 1–4 and Theorem 1. We prove each supporting lemma in a separate subsection later. In Lemmas 5–7, we establish some relations between PC and E. In Lemma 5, we show that successful transmissions in E achieve the required QoS if such is achieved in PC. In Lemmas 6 and 7, we stochastically bound the number of packet drops in E. Thus, most of the packets achieve the required QoS in E. Finally, in Lemma 8, we relate the performances in E and EC.

Lemma 5: If all the packets are successfully transmitted before their respective finish times in PC, then all the packets that are successfully transmitted in E also depart before their respective finish times in E.

Lemma 6: Let n_i and $\widehat{n}_i^E$ denote the number of packets generated by session i and the number of session i packets that cannot be transmitted successfully before their finish times in E, respectively. Then, $\mathbf{E}[\widehat{n}_i^E/n_i] \leq \mu_i$ if 1) the delay $D_j - V$ is guaranteeable in PC for every admitted session j and 2) the probability that the number of channel errors of i exceeds V_i in any busy period in E is at most μ_i.

Lemma 7: Let

$$Z = \min_{t\geq 0}\left\{t \;:\; \sum_{j\in\mathcal{C}} g_j(t) \leq \left(1-\sum_{j\in\mathcal{C}}\delta_j\right)t\right\} \tag{3}$$

and

$$\delta_i = \frac{\log(\beta_i)}{\log\left(\frac{\beta_i}{\zeta_i}\right)} + \frac{\log\left(\frac{\alpha_i}{\min(\mu_i,\alpha_i)}\right)}{\widehat{Z}\log\left(\frac{\beta_i}{\zeta_i}\right)} \tag{4}$$

where

$$\widehat{Z} = \min_{t\geq 0}\left\{t \;:\; \sum_{j\in\mathcal{C}} g_j(t) \leq \left(1-\sum_{j\in\mathcal{C}}\frac{\log(\beta_j)}{\log\left(\frac{\beta_j}{\zeta_j}\right)}\right)t\right\}. \tag{5}$$

Then, the probability that the number of channel errors of an admitted session i in a busy period in E exceeds $V_i = \delta_i Z$ is at most μ_i.

Note that $\alpha_i, \beta_i, \zeta_i$ have been defined in Fig. 4, and $g_i(t)$ has been defined in Section III.

The phrase "beginning of a slot" refers to 1) the instant just before the transmission of a packet, if a packet is transmitted in the slot, and 2) the instant just after the slot starts otherwise.

Lemma 8:

C1) At the beginning of slot t, a packet $p_{k,i}$ has the smallest finish time in E iff $p_{k,i}$ has the smallest finish time among the packets in primary queues in EC.

C2) At the beginning of slot t, the value of compensation counter is the same in E and EC for every admitted session.

C3) If E transmits a packet $p_{k,i}$ successfully at slot t, then EC transmits a packet $p_{k_1,i}$ successfully at t, where $k_1 \leq k$. If $k_1 < k$, $p_{k_1,i}$ is transmitted from i's compensation queue.

C4) A packet $p_{k,i}$ departs from system E (either transmitted successfully or dropped) at slot t iff the packet departs i's primary queue (either transmitted successfully or transferred to i's compensation queue) in EC at slot t.

PROOFS OF RESULTS IN SECTION VI

In this section, we prove the performance guarantees provided by the joint admission control and scheduling framework (Figs. 4 and 5). First, we present a lemma that is used in many of the following proofs.

Lemma 9:

O1) In PC, every packet $p_{k,i}$ is successfully transmitted before its finish time $f_{k,i}^{PC}$.

O2) In EC, the deadline for every packet $p_{k,i}$ is greater than or equal to the packet's finish time $f_{k,i}^{EC}$ plus V.

Proof: The admission control algorithm (Fig. 5) ensures that $D_i - V$ is guaranteeable in PC for every admitted session i. Thus, O1) follows from Definition 1. Also, from Definition 1, every packet $p_{k,i}$ satisfies $f_{k,i}^{PC} \leq a_{k,i} + D_i - V$. Now, O2) follows from A1) and (2). ■

A. Proof of Lemma 1

Proof: In EC, let $p_{k,i}$ be transmitted successfully at slot t. If $p_{k,i}$ is transmitted from i's compensation queue, then the scheduling ensures that the deadline for the packet is greater than t. If $p_{k,i}$ is transmitted from i's primary queue at t, then $p_{k,i}$ has the smallest finish time among the packets present in the primary queues. From C1) in Lemma 8, $p_{k,i}$ has the smallest finish time in E at t. Thus, it is transmitted in E. The channel state for session i is "good" at t as the packet transmission is successful in EC at t. So, the packet transmission is successful in E as well. Thus, $p_{k,i}$ departs E before its finish time in E [Lemma 5 and O1) in Lemma 9]. Thus, $t \leq f_{k,i}^E$. Now, the result follows from (2) and O2) in Lemma 9. ■

B. Proof of Lemma 2

Proof: Consider EC. The number of i's compensation slots in a primary busy period is always less than or equal to i's channel errors in the busy period (Definition 3). Thus, from the condition given in the lemma, the number of i's compensation slots is upper bounded by V_i in every primary busy period. Now, if a compensation queue is empty at the beginning of a primary busy period, then a packet for session i is transferred from the primary queue to the compensation queue only if the number of i's compensation slots is greater than V_i in the primary busy period (Fig. 5). Since the compensation queue for session i is empty at the beginning of the first primary busy period, it can be seen using an induction argument that none of i's packets are transferred to its compensation queue. Thus, each of i's packets is transmitted from i's primary queue. Since packets are never dropped from the primary queue (Fig. 5), we conclude that all of i's packets are transmitted successfully. Let one such packet $p_{k,i}$ be transmitted successfully at t. Now, since i's packets are never transferred to its compensation queue, from A3) and C4) in Lemma 8, $p_{k,i}$ is transmitted successfully in E at slot t. Now, from O1) in Lemma 9, Lemma 5, and (2), $f_{k,i}^{EC} + V \geq t$. The result follows. ■

C. Proof of Lemma 3

Proof: From A1) and C4) in Lemma 8, it follows that the busy periods in E and the primary busy periods in EC coincide, i.e., if the vth busy period in E starts at b_v and finishes at B_v, then so does the vth primary busy period in EC for every v. Hence, if the probability that the number of i's channel errors exceeds V_i in a primary busy period in EC is at most μ_i, then the probability that the number i's channel errors exceeds V_i in a busy period in E is at most μ_i. From the admission control algorithm, $\vec{D} - V$ is guaranteeable in PC. Thus, from Lemma 6, the expected fraction of i's packets dropped in E is at most μ_i. From C3) in Lemma 8, the number of i's packets transmitted successfully in EC is greater than or equal to that in E. The result follows. ■

D. Proof of Theorem 1

Proof: From A1), E and EC have the same set of sessions, and from A3), the channel state for every session is the same in both systems. From Lemma 7, the probability that the number of i's channel errors in a busy period in E exceeds V_i is at most μ_i. From C4) in Lemma 8, the busy periods in E and the primary busy periods in EC coincide. Hence, the probability that the number of i's channel errors in a primary busy period in EC exceeds V_i is also at most μ_i. The result follows from Lemma 3. ■

E. Proof of Lemma 4

Proof: Let $k_1 < k_2$. We prove that at any slot t, if $p_{k_2,i}$ is successfully transmitted in EC, then $p_{k_1,i}$ has already been successfully transmitted or dropped. The packets depart the primary and compensation queues in increasing order of their arrivals, i.e., if p_1 enters i's primary (compensation) queue before p_2, then p_1 departs from the primary (compensation) queue before p_2 (Fig. 5). First, let $p_{k_2,i}$ be transmitted from i's primary queue at slot t. Then, $p_{k_2,i}$ is at the HoL position in i's primary queue, and i's compensation queue is empty at t after expunging i's packets whose deadlines have expired. Thus, $p_{k_1,i}$ is not in the system at t. Next, let $p_{k_2,i}$ be transmitted from i's compensation queue at t. Then, $p_{k_2,i}$ is the HoL packet in i's compensation queue after expunging i's packets whose deadlines have expired. Thus, $p_{k_1,i}$ is not in i's compensation queue, and not in i's primary queue, since $p_{k_2,i}$ has already left the primary queue. Thus, again, $p_{k_1,i}$ is not in the system at t. The result follows. ■

Proofs for Supporting Lemmas

A. Proof of Lemma 5

Proof: We first state a relation that is a direct consequence of (2) and the definition for $w_i[t, t+\tau]$, and a result from [43], [44], i.e.,

$$w_i^{PC}[t, t+\tau] = w_i^E[t, t+\tau+V] \; \forall \, i \in \mathcal{C}. \tag{6}$$

Result From [43], [44]: Let $W^{PC}[t, t+\tau]$ represent the maximum value of $w^{PC}[t, t+\tau]$ in any arrival pattern that conforms to the traffic characterization. If all the packets arriving in PC depart before or at their respective finish times, then

$$W^{PC}[t, t+\tau] \leq \tau + 1 \; \forall \, t \text{ and } \tau \geq 0. \tag{7}$$

We prove Lemma 5 using contradiction. Let there exist a packet $p_{k,i}$ such that $p_{k,i}$ is transmitted successfully after its finish time in E, i.e., $d_{k,i}^E > f_{k,i}^E$. Let $T_{k,i}$ denote the starting epoch of the busy period in E in which $p_{k,i}$ arrives. We define $\hat{T}_{k,i}$ to be the smallest time instant such that

T1) $\hat{T}_{k,i} \in [T_{k,i}, a_{k,i}]$;

T2) no packet with finish time greater than $f_{k,i}^E$ is transmitted in the interval $[\hat{T}_{k,i}, d_{k,i}^E]$.

Note that $\hat{T}_{k,i}$ always exists. This is because the packet with the smallest finish time is transmitted in every slot in E, and between $a_{k,i}$ and $d_{k,i}^E$, the packet $p_{k,i}$ with finish time $f_{k,i}^E$ is present in the system. Thus, no packet with finish time greater than $f_{k,i}^E$ is transmitted in $[a_{k,i}, d_{k,i}^E]$. Thus, $\hat{T}_{k,i} \leq a_{k,i} \leq f_{k,i}^{PC} \leq f_{k,i}^E$. The second inequality follows since all packets of all admitted sessions must depart before their finish times in PC, and hence, their arrival times cannot exceed their finish times in PC. The last inequality follows from (2).

Now, we show that $w^E[\hat{T}_{k,i}, f_{k,i}^E] \geq \widehat{S}^E[\hat{T}_{k,i}, f_{k,i}^E]$. From the property T2) in the definition of $\hat{T}_{k,i}$, and since $\hat{T}_{k,i} \leq f_{k,i}^E < d_{k,i}^E$, the packets transmitted in $[\hat{T}_{k,i}, f_{k,i}^E]$ have finish times less than or equal to $f_{k,i}^E$. Using contradiction, we now show that these packets also arrive in $[\hat{T}_{k,i}, f_{k,i}^E]$. It follows that $w^E[\hat{T}_{k,i}, f_{k,i}^E] \geq \widehat{S}^E[\hat{T}_{k,i}, f_{k,i}^E]$. Let a packet $p_{u,j}$ arrive before $\hat{T}_{k,i}$ and be transmitted at $t \in [\hat{T}_{k,i}, f_{k,i}^E]$. Thus, $f_{u,j}^E \leq f_{k,i}^E$. Since E always transmits packets in the increasing order of their finish times, no packet with finish time greater than $f_{u,j}^E$ can be transmitted in $[a_{u,j}, t]$. Thus, no packet with finish time greater than $f_{k,i}^E$ is transmitted in $[a_{u,j}, t]$. From the definition of $\hat{T}_{k,i}$ and since $t \in [\hat{T}_{k,i}, f_{k,i}^E]$, we conclude that $\hat{T}_{k,i} \leq a_{u,j}$, which is a contradiction.

Now, since $p_{k,i}$ contributes to $w^E[\hat{T}_k, f_{k,i}^E]$ but departs after $f_{k,i}^E$, $w^E[\hat{T}_{k,i}, f_{k,i}^E] > \widehat{S}^E[\hat{T}_{k,i}, f_{k,i}^E]$. In E, if in a busy period the number of session j's compensation slots exceeds V_j, then each subsequent packet of j is removed from the queue after one transmission. Recall that $V = \sum_{j \in \mathcal{C}} V_j$. Thus, $\widehat{S}^E[\hat{T}_{k,i}, f_{k,i}^E] \geq f_{k,i}^E - \hat{T}_{k,i} + 1 - V$. Thus, $w^E[\hat{T}_k, f_{k,i}^E] > f_{k,i}^E - \hat{T}_{k,i} + 1 - V$.

Now, from (2) and (6)

$$W^{PC}\left[\hat{T}_{k,i}, f_{k,i}^{PC}\right] \geq w^{PC}\left[\hat{T}_{k,i}, f_{k,i}^{PC}\right] > f_{k,i}^{PC} - \hat{T}_{k,i} + 1.$$

Then, from (7), all the packets do not depart before their respective finish times in PC. This is a contradiction. ■

B. Proof of Lemma 6

Proof: Consider system E. From Lemma 5 and O1) in Lemma 9, all the packets that are successfully transmitted depart before their finish times. Hence, for every $j \in \mathcal{C}$, $\widehat{n}_j^E$ is equal to the number of packets dropped for session j.

Consider a busy period $\mathcal{B}$ and a session i packet P that arrives during $\mathcal{B}$. Now, let $\mathcal{A}$ denote the event that a packet from session i is dropped in $\mathcal{B}$. Now, packets for session i are dropped in $\mathcal{B}$ only if the number of i's channel errors in $\mathcal{B}$ exceeds V_i. Thus, $\mathbf{P}\{\mathcal{A}\} \leq \mu_i$. Thus

$$\mathbf{P}\{\text{Packet } P \text{ is dropped}\} \leq \mathbf{P}(\mathcal{A}) \leq \mu_i. \tag{8}$$

Let $\mathbf{1}_{k,i}$ denote an indicator random variable that is equal to 1 if $p_{k,i}$ is dropped and is 0 otherwise. Since P and $\mathcal{B}$ are arbitrary, from (8), we conclude that for every $k \in \{1, \ldots, n_i\}$

$$\mathbf{E}[\mathbf{1}_{k,i}] \leq \mu_i. \tag{9}$$

Now, the fraction of packets dropped is $\widehat{n}_i^E/n_i = \sum_{k=1}^{n_i} \mathbf{1}_{k,i}/n_i$. From (9), $\mathbf{E}[\widehat{n}_i^E]/n_i \leq \mu_i$. ■

C. Proof of Lemma 7

Let T be a constant. We obtain a function $\phi_i(T)$ such that the probability that the number of channel errors of i in T slots exceeds $T\phi_i(T)$ is at most μ_i (Lemmas 10–12). We prove Lemma 7 using the properties of $\phi_i(T)$.

Lemma 10: Let the channel for session i have a "good" state in slot 0. Let $X_{k,i}$ be a random variable that represents the number of slots between $k-1$th and kth visits of i's channel to "good" state. Then, $\mathbf{P}\{\sum_{k=1}^{n} X_{k,i} \geq na\} \leq (\beta_i \zeta_i^{a-1})^n$.

Proof: $X_{k,i}$ has the distribution

$$\mathbf{P}\{X_{k,i} = 1\} = p_i$$
$$\mathbf{P}\{X_{k,i} = u\} = (1-p_i)(1-q_i)q_i^{(u-2)} \ \forall\, u \geq 2.$$

Thus, $\forall \theta \in [0, \ln(1/q_i))$

$$\mathbf{E}[e^{\theta X_{k,i}}] = \frac{p_i e^{\theta} - (p_i + q_i - 1)e^{2\theta}}{1 - q_i e^{\theta}}. \tag{10}$$

Since the channel errors constitute a Markov process, the random variables $X_{k,i}$'s are independent. Now

$$\begin{aligned}
&\mathbf{P}\left\{\sum_{k=1}^{n} X_{k,i} \geq na\right\} \\
&\leq \inf_{\theta \geq 0} \left[e^{-n\theta a}\mathbf{E}\left(e^{\theta \sum_{k=1}^{n} X_{k,i}}\right)\right] \\
&= \inf_{\theta \geq 0} e^{-n\theta a}\Pi_{k=1}^{n}\mathbf{E}(e^{\theta X_{k,i}}) \ \text{(since } X_{k,i} \text{ are independent)} \\
&\leq \left[e^{-\theta a}\frac{p_i e^{\theta} - (p_i + q_i - 1)e^{2\theta}}{1 - q_i e^{\theta}}\right]^n \ \forall\, \theta \in \left[0, \ln\frac{1}{q_i}\right).
\end{aligned} \tag{11}$$

Now, (11) follows from (10). The lemma follows by using $\theta = \ln(1 + q_i/2q_i)$, and thereafter, $\beta_i = (1 + q_i - p_i/q_i)$ and $\zeta_i = (2q_i/1 + q_i)$ in (11). ■

Let $Y_{k,i}$ be an r.v. indicating the number of slots between the $k-1^{\text{th}}$ and the k^{th} visits to state "good."

Lemma 11: Let the MC indicating the channel state of session i be in steady state. Then

$$\mathbf{P}\left\{\sum_{k=1}^{n} Y_{k,i} \geq na\right\} \leq \alpha_i \left[\beta_i \zeta_i^{a-1}\right]^n. \tag{12}$$

Note that when a busy period starts, session i's channel is in "good" ("bad") state w.p. $\pi_{G,i}$ ($\pi_{B,i}$). Lemma 11 follows by conditioning on the channel state of session i at the beginning of a typical busy period and by using arguments similar to that in Lemma 10. We prove Lemma 11 in the technical report in [45].

Lemma 12: Consider any interval of T slots and a constant c. Let $\mathcal{A}_i$ denote the event that the number of visits to "bad" state for session i in T slots is greater than cT. Then, $\mathbf{P}(\mathcal{A}_i) \leq \mu_i$ if $c \geq \phi_i(T)$, where

$$\phi_i(T) = \frac{\log(\beta_i)}{\log\left(\frac{\beta_i}{\zeta_i}\right)} + \frac{\log\left(\frac{\alpha_i}{\min(\mu_i, \alpha_i)}\right)}{T\log\left(\frac{\beta_i}{\zeta_i}\right)}. \tag{13}$$

Proof: The result clearly holds when $c \geq 1$. Let $c < 1$. Since $c \geq \phi_i(T)$, from (13), $(\log(\beta_i)/\log(\beta_i/\zeta_i)) + (\log(\alpha_i/\min(\mu_i, \alpha_i))/T\log(\beta_i/\zeta_i)) \leq c$. Thus

$$\alpha_i \beta_i^{T(1-c)} \zeta_i^{cT} \leq \min(\mu_i, \alpha_i) \leq \mu_i. \tag{14}$$

Now, $\sum_{k=1}^{(1-c)T} Y_{k,i} \geq T$ iff the time required for $(1-c)T$ visits to "good" state is greater than or equal to T, which in turn happens iff event $\mathcal{A}_i$ occurs. Let $n = (1-c)T$ and $a = (1/1 - c)$. Thus, $\mathbf{P}(\mathcal{A}_i) = \mathbf{P}\{\sum_{k=1}^{n} Y_k \geq na\}$. Now, from Lemma 11 and since $n = T/a$, $\mathbf{P}(\mathcal{A}_i) \leq \alpha_i(\beta_i \zeta_i^{a-1})^{T/a} \leq \mu_i$, where the last inequality follows from (14). ■

Now, we prove Lemma 7.

Proof: From (13), $\phi_i(T) \leq \phi_i(\widehat{T})$, whenever $T \geq \widehat{T}$. Now, from (3)–(5), $Z \geq \widehat{Z}$. Hence

$$\phi_i(Z) \leq \phi_i(\widehat{Z}). \tag{15}$$

Let $\mathcal{A}_i$ denote the event that the number of channel errors for session i in Z slots exceeds $Z\phi_i(\widehat{Z})$. From (15) and Lemma 12, it follows that $\mathbf{P}(\mathcal{A}_i) \leq \mu_i$. Note that $V_i = \delta_i Z = \phi_i(\widehat{Z})Z$.

Consider system E. Now we show that Z is an upper bound on the length of busy period when $V_j = \delta_j Z$ for every j. Now, let there exist a busy period that has length greater than Z. Without loss of generality, let this busy period begin at slot 0. Thus, the number of packets that arrived till Z is greater than or equal to number of packets that departed till Z. Thus

$$\sum_{j \in \mathcal{C}} \gamma_j[0, Z] > \widehat{S}^E[0, Z]. \tag{16}$$

Now, if the number of compensation slots for session i in a busy period is greater than or equal to V_j, then packets of session j depart the system when transmitted. Hence

$$\widehat{S}^E[0, Z] \geq Z - \sum_{j \in \mathcal{C}} V_j = \left(1 - \sum_{j \in \mathcal{C}} \delta_j\right) Z. \tag{17}$$

Furthermore, by definition

$$\sum_{j \in \mathcal{C}} \gamma_j[0, Z] \leq \sum_{j \in \mathcal{C}} g_j(Z). \tag{18}$$

From (16)–(18), we conclude that $\sum_{j \in \mathcal{C}} g_j(Z) > (1 - \sum_{j \in \mathcal{C}} \delta_j)Z$. This contradicts (3). Hence, Z is an upper bound on the length of the busy period.

Hence, if the number of channel errors for session i in a busy period exceeds V_i, then event $\mathcal{A}_i$ occurs since $V_i = \phi_i(\hat{Z})Z$. The result follows since $\mathbf{P}(\mathcal{A}_i) \leq \mu_i$. ■

D. *Proof of Lemma 8*

Proof: We prove Lemma 8 using induction on t.

Base Case $t = 0$: The systems E and EC have no packet queued before slot 0. Hence, if no packet arrives at $t = 0$, then C1) to C4) hold by vacuity. Now, let packets arrive at slot 0.

Proof of C1) at slot 0: The arriving packets are queued in E and in the primary queues in EC. Thus, C1) follows from (2).

Proof of C2) at slot 0: Since a busy period in E and a primary busy period in EC start at slot 0, before any transmission at 0, the compensation counter for every session j is initialized to 0 in both systems. Thus, C2) follows.

Consider slot 0. E transmits a packet $p_{k,i}$ iff $p_{k,i}$ has the smallest finish time in E. From C1), the latter happens iff $p_{k,i}$ has the smallest finish time among all the packets in the primary queues in EC. Now, since the compensation queues in EC are empty before any transmission, the above happens iff $p_{k,i}$ is transmitted in EC.

Proof of C3) at slot 0: Now, if $p_{k,i}$ is successfully transmitted in E at slot 0, then i's channel is in good state at 0, which in turn implies that $p_{k,i}$ is successfully transmitted in EC at 0. Thus, C3) follows.

Proof of C4) at slot 0: Consider slot 0. A packet $p_{k,i}$ departs from E (EC) iff it is transmitted in E (EC) and either 1) the channel for session i is "bad," and before the transmission i's compensation counter in E (EC) equals V_i, or 2) the channel for i is "good." Now, $p_{k,i}$ is transmitted in E iff $p_{k,i}$ is transmitted in EC. The channel for i has the same state in both E and EC. Also, from C2) before transmission, i's compensation counter in E equals that in EC. Thus, C4) follows.

Induction Hypothesis (IH): Statements C1) to C4) hold at t if $t \in \{0, \dots, T\}$.

Note that the packets in E and the packets in the primary queues of EC are the same at the end of any slot $t \leq T$. This follows from C4) of IH and since the same packets arrive in E and the primary queues of EC.

Proof of C1) at slot $T+1$: Again, the same packets arrive at $T+1$ in both E and the primary queues of EC. Thus, at the beginning of $T+1$, the same packets are present in E and in the primary queues of EC. Now, C1) follows from (2).

Consider slot t, where $t \in \{0, \dots, T+1\}$. Let $p_{k,i}$ be transmitted in E. Thus, $p_{k,i}$ has the smallest finish time in E. From C1), $p_{k,i}$ has the smallest finish time among all the packets in the primary queues in EC. Thus, EC transmits some packet of session i, $p_{k,i}$, or a packet in i's compensation queue.

Proof of C2) at slot $T+1$: We first assume that at least one queue has a packet at slot T in E. Thus, a packet is transmitted in E, and by the argument in the previous paragraph, the same session (say i) transmits a packet in EC. Let i's channel be "good" at T. Then, the transmissions in E and EC are successful, and the values of the compensation counters do not change in T in both systems. Now, let i's channel be "bad" at T. Then, the transmissions in E and EC are not successful, and subsequently, i's compensation counters are incremented by one in both systems. Clearly, C2) follows at $T+1$ from IH at T in both cases. We now assume that the queues for all the sessions in E are empty at slot T. Then, the primary queues for all the sessions in EC are also empty at T. Thus, no packet is transmitted in E, and no packet is transmitted from the primary queues in EC at t. Thus, the compensation counters do not change in these systems in T. Now, if no packet arrives in $T+1$, the compensation counter for each session is the same at the beginning of $T+1$ as that at the beginning of T in both E and EC. Thus, C2) follows from IH. If packets arrive in $T+1$, then a busy period in E and a primary busy period in EC start at $T+1$. Hence, the value of the compensation counter for every session i is initialized to 0 at the beginning of $T+1$ in both systems. Thus, C2) follows.

Proof of C3) at slot $T+1$: Consider slot $T+1$. Let $p_{k,i}$ be transmitted successfully in E. Thus, session i transmits a packet (say $p_{k_1,i}$) in EC, and i's channel is in good state. Thus, $p_{k_1,i}$ is transmitted successfully. We consider two cases, namely a) no packet in i's compensation queue has a valid deadline, and b) a packet in i's compensation queue has a valid deadline. In (a), i transmits a packet from its primary queue, and this packet is $p_{k,i}$ as $p_{k,i}$ has the minimum finish time in i's primary queue. Thus, $k_1 = k$. In (b), $p_{k_1,i}$ is in i's compensation queue. Thus, since $p_{k,i}$ is in i's primary queue, $p_{k_1,i}$ has arrived before $p_{k,i}$. Hence, $k_1 < k$. Thus, C3) holds at $T+1$.

Proof of C4) at slot $T+1$: Consider slot $T+1$. Let $p_{k,i}$ depart from E. Thus, $p_{k,i}$ is transmitted in E, and either a) the channel for session i is "bad," and before the transmission, i's compensation counter in E equals V_i, or b) the channel for i is "good." Session i transmits a packet in EC, and $p_{k,i}$ has the minimum finish time among the packets in the primary queues in EC. Also, from C2), before the transmission, the compensation counters have the same value for every session in E and EC. Now, under both (a) and (b), $p_{k,i}$ departs i's primary queue in EC. Using similar arguments, we can prove that if $p_{k,i}$ depart i's primary queue in EC at $T+1$, $p_{k,i}$ departs E at $T+1$. Thus, C4) holds at $T+1$.

Hence, by induction, C1) to C4) hold at every slot. ■

REFERENCES

[1] J. Kurose and K. Ross, *Computer Networking: A Top-Down Approach Featuring the Internet*, 2nd ed. Reading, MA: Addison-Wesley, 2003.

[2] V. Bharghavan, S. Lu, and T. Nandagopal, "Fair queueing in wireless networks: Issues and approaches," *IEEE Pers. Commun. Mag.*, vol. 6, no. 1, pp. 44–53, Feb. 1999.

[3] T. S. E. Ng, I. Stoica, and H. Zhang, "Packet Fair Queueing Algorithms for Wireless Networks With Location Dependent Errors," Carnegie Mellon Univ., Pittsburgh, PA, Tech. Rep. CMU-CS-00-112, 2000.

[4] M. Krunz and J. G. Kim, "Fluid analysis of delay and packet discard performance for QoS support in wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 19, no. 2, pp. 384–395, Feb. 2001.

[5] S. Wang, R. Nathuji, R. Bettati, and W. Zhao, "Providing statistical delay guarantees in wireless networks," in *Proc. ICDCS*, Mar. 2004, pp. 48–55.

[6] S. Ayyorgun and R. L. Cruz, "A composable service model with loss and a scheduling algorithm," in *Proc. INFOCOM*, 2004, pp. 1950–1961.

[7] ——, "A service curve model with loss and multiplexing problem," in *Proc. ICDCS*, 2004, pp. 756–765.

[8] S. Lu, V. Bharghavan, and R. Shrikant, "Fair scheduling in wireless packet networks," in *Proc. ACM SIGCOMM*, 1997, pp. 63–74.

[9] S. Lu, T. Nandagopal, and V. Bharghavan, "Design and analysis of an algorithm for fair service in error-prone wireless channels," *Wirel. Netw.*, vol. 6, no. 4, pp. 323–343, 2000.

[10] S. Shakkottai and A. Stolyar, "Scheduling for multiple flows sharing a time-varying channel: The exponential rule," in *Translations of the AMS, A Volume in Memory of F. Karpelevich*. Providence, RI: American Mathematical Society, 2001.

[11] L. Tassiulas and A. Ephremides, "Dynamic server allocation to parallel queue with randomly varying connectivity," *IEEE Trans. Inf. Theory*, vol. 30, no. 2, pp. 466–478, Mar. 1993.

[12] M. Goyal, A. Kumar, and V. Sharma, “Power constrined and delay optimal policies for scheduling transmission over a fading channel,” in *Proc. INFOCOM*, 2003, pp. 311–320.
[13] S. Shakkottai and R. Shrikant, “Scheduling real time traffic with deadlines over a wireless channel,” *ACM/Beltzer Wirel. Netw. J.*, vol. 8, no. 1, pp. 13–26, 2001.
[14] R. Jain and E. W. Knightly, “A framework for design and evaluation of admission control algorithms in multi-service mobile networks,” in *Proc. INFOCOM*, 1999, pp. 1027–1035.
[15] D. Julian, M. Chiang, D. O’Neill, and S. Boyd, “QoS and fairness constrained convex optimization of resource allocation for wireless cellular and ad hoc networks,” in *Proc. INFOCOM*, 2002, pp. 477–486.
[16] B. Sadegi, “Architecture and algorithms for scalable mobile QoS,” M.S. thesis, Rice Univ., Houston, TX, 2000.
[17] D. Shen and C. Ji, “Admission control of multiple traffic in third generation CDMA networks,” in *Proc. INFOCOM*, 2000, pp. 1077–1086.
[18] ——, “Capability based admission control for broadband CDMA networks,” in *Proc. VTC*, 2001, pp. 202–206.
[19] C. Chao and W. Chen, “Connection admission control for mobile multiple-class personal communications networks,” *IEEE J. Sel. Areas Commun.*, vol. 15, no. 8, pp. 1618–1626, Oct. 1997.
[20] D. Levine, I. Akyildiz, and M. Naghshineh, “A resource estimation and call admission control algorithm for wireless multimedia networks using the shadow cluster concept,” *IEEE/ACM Trans. Netw.*, vol. 5, no. 1, pp. 1–12, Feb. 1997.
[21] A. Talukdar, B. Badrinath, and A. Acharya, “On accommodating mobile hosts in an integrated services packet networks,” in *Proc. INFOCOM*, 1997, pp. 1046–1053.
[22] A. Acampora and M. Naghshineh, “An architecture and methodology for mobile-executed handoff in cellular ATM networks,” *IEEE J. Sel. Areas Commun.*, vol. 12, no. 8, pp. 1365–1375, Oct. 1994.
[23] ——, “QoS provisioning in micro-cellular networks supporting multimedia traffic,” in *Proc. INFOCOM*, 1995, pp. 1075–1084.
[24] A. Aljadhai and T. Zhanti, “A framework for call admission control and QoS support in wireless environments,” in *Proc. INFOCOM*, 1999, pp. 1019–1026.
[25] P. Bhagwat, A. Krishna, and S. Tripathi, “Enhancing throughput over wireless LANs using channel state dependent packet scheduling,” in *Proc. INFOCOM*, 1996, pp. 1133–1140.
[26] S. Shakkottai and A. L. Stolyar, “Scheduling algorithm for a mixture of real-time and non-real-time data in HDR,” in *Proc. ITC*, 2001, pp. 793–804.
[27] R. L. Cruz, “A calculus for network delay—Part I: Network elements in isolation,” *IEEE Trans. Inf. Theory*, vol. 37, no. 1, pp. 114–131, Jan. 1991.
[28] E. N. Gilbert, “Capacity of burst-noise channel,” *Bell Syst. Tech. J.*, vol. 39, no. 8, pp. 1253–1265, Sep. 1960.
[29] E. O. Elliaot, “Estimation of error rates for codes on burst-noise channels,” *Bell Syst. Tech. J.*, vol. 42, no. 9, pp. 1977–1997, Sep. 1963.
[30] J. G. Kim and M. Krunz, “Bandwidth allocation in wireless networks with guaranteed packet loss performance,” *IEEE Trans. Veh. Technol.*, vol. 49, no. 5, Sep. 2000.
[31] C. L. Liu and J. W. Layland, “Scheduling algorithms for multiprogramming in a hard real time environment,” *J. ACM*, vol. 20, no. 1, pp. 46–61, Jan. 1973.
[32] 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. Netw.*, vol. 1, no. 3, pp. 344–357, Jun. 1993.
[33] A. Demers, S. Keshav, and S. Shenkar, “Analysis and simulation of a fair queueing algorithm,” *Internetw. Res. Exp.*, vol. 1, no. 1, pp. 3–26, 1990.
[34] S. J. Golestani, “A self-clocked fair queueing scheme for high speed applications,” in *Proc. INFOCOM*, Apr. 1994, pp. 636–646.
[35] L. Zhang, “Virtualclock: A new traffic control algorithm for packet switching networks,” in *Proc. ACM SIGCOMM*, Aug. 1990, pp. 19–29.
[36] J. C. R. Bennett and H. Zhang, “wf^2q : Worst-case fair weighted fair queueing,” in *Proc. INFOCOM*, Mar. 1996, pp. 120–127.
[37] P. Goyal and H. M. Vin, “Generalized guaranteed rate scheduling algorithms: A framework,” *IEEE/ACM Trans. Netw.*, vol. 5, no. 4, pp. 561–571, Aug. 1997.
[38] D. Stiliadis and A. Varma, “Rate proportional servers: A design methodology for fair queueing algorithms,” *IEEE/ACM Trans. Netw.*, vol. 6, no. 2, pp. 164–174, Apr. 1998.
[39] H. Sariowan, R. L. Cruz, and G. C. Polyzos, “Sced: A generalized scheduling policy for guaranteeing quality-of-service,” *IEEE/ACM Trans. Netw.*, vol. 7, no. 5, pp. 669–684, Oct. 1999.
[40] H. Zhang, “Service desciplines of guaranteed performance service in packet-switching networks,” *Proc. IEEE*, vol. 83, no. 10, pp. 1374–1396, Oct. 1995.
[41] R. Agrawal and R. Rajan, “Performance bounds for guaranteed and adaptive services,” IBM, Yorktown Heights, NY, Tech. Rep. RC 20649, 1996.
[42] J. Y. Le Boudec and G. Hebuterne, “Application of network calculus to guaranteed service networks,” *IEEE/ACM Trans. Netw.*, vol. 8, no. 3, 1998.
[43] H. Sariowan, “A service-curve approach to performance guarantees in integrated-service networks,” Ph.D. dissertation, Univ. California, San Diego, La Jolla, CA, 1996.
[44] P. Chaporkar, “An approach to the analysis of scheduling policies for guaranteeing delay with arbitrary arrivals,” M.S. thesis, Indian Inst. Sci., Bangalore, India, 2000. [Online]. Available: http://shravana.cedt.iisc.ernet.in/~scprasan
[45] P. Chaporkar and S. Sarkar, “Admission control framework to provide guaranteed delay in error-prone wireless channel,” Univ. Pennsylvania, Philadelphia, PA, 2002. Tech. Rep., [Online]. Available: http://www.seas.upenn.edu/~swati
[46] Q. Zhang and S. A. Kassam, “Finite-state Markov model for Rayleigh fading channel,” *IEEE Trans. Commun.*, vol. 47, no. 11, pp. 1688–1692, Nov. 1999.

Prasanna Chaporkar received the B.E. degree in electronics from Walchand College of Engineering, Maharashtra, India, in 1996, the M.S. degree from the Faculty of Engineering, Indian Institute of Science, Bangalore, India, in 2000, and the Ph.D. degree from the University of Pennsylvania, Philadelphia, in 2006. He is currently pursuing a post-doctoral investigation with INRIA, Paris, France.

His research interests are in resource allocation in communication networks, stochastic control, and queueing theory.

Saswati Sarkar (S’98–M’00) received the Master of Engineering degree in electrical communication engineering from the Indian Institute of Science, Bangalore, India, in 1996 and the Ph.D. degree in electrical and computer engineering from the University of Maryland, College Park, in 2000.

She is currently an Assistant Professor with the Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia. Her research interests are in resource allocation and performance analysis in communication networks.

Dr. Sarkar has been an Associate Editor of IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS since 2001. She received the Motorola Gold Medal for the best masters student from the Division of Electrical Sciences, Indian Institute of Science, and the National Science Foundation (NSF) Faculty Early Career Development Award in 2003.