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

Prasanna Chaporkar and Saswati Sarkar
Department of Electrical and Systems Engineering
University of Pennsylvania
{prasanna@seas, swati@ee}.upenn.edu

***Abstract*— An admission control mechanism is a key component of any framework that provides guaranteed delay service to heterogeneous real-time traffic. In this paper, we propose a general framework for admission control to provide delay guarantees in the error-prone wireless systems. By "general" we mean that the scheduling policies from a large class can be plugged in this framework and an admission control condition can be obtained for general arrival processes. The generality of the framework will enable us to use many scheduling policies that have not been considered yet for the error-prone wireless systems. The designed framework only assumes statistical knowledge of channel state and instantaneous knowledge of channel state is not required. Apart from providing delay guarantees in the presence of channel errors, the proposed framework provides (1) a good balance of packet drop and session blocking (2) isolation in the sense that once a session is admitted, then its guarantees (delay and packet drop) are not affected by other sessions.**

I. Motivation

The next generation wireless packet networks will have to provide Quality of Service (QoS) guarantees to heterogeneous real-time traffic. In particular, it needs to provide stringent delay, delay jitter and packet loss guarantees. The key components of any framework that provides guaranteed delay service are (1) an *admission control* algorithm on control path and (2) a *packet scheduling* on the data path.

The admission control ensures that enough resources are available to provide required delay to an admitted session, while scheduling mechanism distributes the system resources among individual flows in accordance with their requirements. The appropriate choice of one depends on another. Further, delay guarantees can not be provided in general, if either of the components is missing. *Our aim is to provide a general framework for admission control in error-prone wireless system*. By "general" we mean that the scheduling policies from a large class can be plugged in this framework and an admission control condition can be obtained for general arrival processes.

Many admission control algorithms are studied for wireline case. There is no obvious extension of these to the wireless case as the wireless channel exhibits location dependent, bursty channel errors. A bad channel state for a session typically results in an erroneous reception of a data packet. Corresponding packet retransmissions account for additional consumption of resources. Hence, the wireline admission control and scheduling can result in an inefficient utilization of scarce wireless bandwidth and excessive packet drop because of deadline expiry.

We note that since the packets can not be served in the duration of the channel errors, which can be long, the packet drop because of deadline expiry is unavoidable in wireless systems. Fortunately, though the real-time applications are sensitive to delay, they can accommodate some packet loss gracefully. But an admission control algorithm that does not take into account the channel statistics for various sessions can result in excessive packet drop. Packet drop above a certain threshold (which can be different for different applications) can cause unacceptable service degradation. Hence, the aim of the admission control algorithm should be to provide a deterministic upper bound on the delay for the transmitted packet and a probabilistic bound on the packet drop.

To limit the packet drop, an admission control algorithm may reserve excess resources. If these excess reservations are not tightly controlled then it can result in high session blocking and reduce the efficiency of resource utilization. Hence the challenge is to *quantify* these excess reservations.

Another interesting problem that exists in wireless case because of the location dependent channel errors is that the delay guarantees of a session depends on the channel errors seen by other sessions. This implies that the excessive channel errors of one session can penalize other flows heavily. This shows a need of *isolation* among various flows. *We propose a joint admission control and scheduling framework that augments scheduling algorithms to control the packet dropping and administer desired features like isolation*.

The framework developed in this paper provides the following features. (1) *Delay guarantees in the presence of channel errors* (2) *good balance between packet drop and session blocking* (3) *isolation in the sense that once a session is admitted, then its delay and packet drop guarantees are not affected by other existing sessions* (4) *guaranteed performance even when the instantaneous channel state is not known. The proposed framework and analytical bounds requires only the statistical knowledge of the wireless channel for various sessions. We like to note that the knowledge of an instantaneous channel state, if available, can be easily accommodated in the framework to enhance the system performance*. We also present extensive simulation results for EDF [14] and PGPS [1] to show that the framework indeed provides all the desired features.

*The work have been supported in part by NSF grant ANI01-06984.

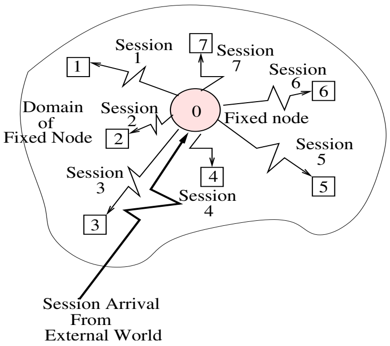


Fig. 1. Sessions from the external node arrive at a fixed node (node 0) in the system. Each session wants to transfer data to one of the destination nodes (nodes 1,2,. . .,7), e.g., Session 1 is transmitting data to node 1.

A. Related Work

Due to space limitations, we have provided a brief literature review. Readers are referred to [2] for further details.

Most of the prior wireless scheduling work [3], [4], [5], [6] obtains delay guarantees for sessions that do not experience channel errors. In [4], [5], authors have obtained the worst case delay bound for the sessions with channel errors. However, these bounds hold only for Head of Line (HoL) packets, in other words no bound has been provided for the overall delay experienced by the packet. Another area extensively explored is that of fair scheduling [7], [8], [9], [10], [11]. The main objective in this branch of research has been to distribute the available resources fairly among the flows [3], [4], [5], [6], [12], which is important for transmitting data traffic. Major concerns for transmission of real time traffic such as deadline expiry and related packet drop have not been addressed though.

In [13], we have studied the admission control problem for EDF scheduler in wireless case. Here, we generalize the results to accommodate a large class of scheduling policies. Also, we propose a scheduling interface that allows us to bound the packet drop for an individual session, which was not possible with the plain EDF scheduling in [13].

The paper is organized as follows. We define our system model in Section II. We present an admission control framework in Section III. In Section IV, we present a scheduling interface to provide isolation in packet drop. In Section V, we present simulation results and discussion.

II. System Model

We consider a single node transmitting to several destinations via error-prone channels. More specifically, we consider a dynamic scenario where sessions arrive at a fixed node and seek to transfer information to some other node in the system (see Figure 1). This fixed node carries the responsibility of admission control and packet scheduling. Hence, we assume that the packet loss is only due to channel errors and not due to collisions or interference. We assume slotted time axis. All packets are assumed to have unit length.

We assume that the arriving sessions specify their traffic and channel characteristics, and the delay requirement. More specifically, every new session i that seeks admission, comes with a three tuple $(\vec{A}_i, \delta_i, D_i)$, where $\vec{A}_i$ signifies a set of parameters for the arrival process, δ_i denotes a long term channel error rate and D_i is a required delay guarantee. We consider deterministic traffic characterizations only. A traffic characterization is deterministic if there exists a monotonically non-decreasing function $f(\tau)$ of time duration τ such that the maximum amount of data that a session can send in the duration τ is less than or equal to the function value.

In the wireless case, the channel capacity varies with time. Further, the channel state for different mobile hosts can be different. So, we model bursty channel errors by the ON-OFF Markov Chain (MC). Thus, δ_i is the steady state probability that the MC for session i is in the OFF state. Instead of sessions specifying δ_i, it can also be that the fixed node knows channel characteristics of its destination nodes. We allow a possibility that sessions with the same destination node have different channel characteristics. This scenario arises in practice when different power control/channel coding schemes are used. Admission control algorithm uses knowledge about the traffic characteristics and the channel statistics of the sessions and a current system load to determine the possibility of providing required delay without violating the delay guarantees of the existing sessions. If it decides that the delay guarantees can be met, then the session is admitted in the system, otherwise it is blocked.

We note that the admission control algorithm requires knowledge of channel behavior typically on a large time scale. These channel statistics can be learned through effective estimation techniques [3]. *But this knowledge does not imply the knowledge of instantaneous channel state. Hence we do not assume the knowledge of channel state at a scheduling instance*. So, packets can be received in error at the destination, which implies a need of retransmission. To decide whether a transmitted packet is received without error, we assume immediate feedback from a receiver.

III. Admission Control Framework

We are considering a dynamic scenario, where sessions arrive one by one and seek admission in a system. Each of the arriving sessions specifies its traffic and channel characteristics, and demands certain delay guarantee from the system. In the dynamic setting, the system may have some existing sessions when a new session arrives. An admission control mechanism should determine, whether a required delay can be provided to a new session without violating delay guarantees of the existing sessions (admission control condition). In the wireless system, it is important to consider channel errors while making this decision. The admission control condition will also depend upon the precise scheduling scheme used. We do not restrict ourselves to any specific scheduling policy, rather our framework allows any dynamic scheduling policy [14] that satisfies ordering property [1], [15]. In this section, we design a general framework for admission control.

Our admission control algorithm is based on the following key result. Let $\vec{D} = [D_1 \ldots D_n]$ be the vector of delay requirements of n active sessions in the system. *We say that the delay vector $\vec{D}$ is guaranteeable if all the packets from every session meet their respective deadline*. Then the following result holds.

Proposition 1: The delay vector $\vec{D}$ is guaranteeable under scheduling policy π in the wireless system if

1) the total channel errors in a busy period are bounded above by V and
2) the delay vector $\vec{D} - V$ is guaranteeable under policy π in the system with error free channel.

We note that if the conditions in Proposition 1 hold, then every packet meets its deadline and consequently there is no packet drop in the system. The main limitation of Proposition 1 is that the channel errors are not bounded in general and hence condition (1) may not hold. On the other hand, the above proposition says that the probability of packet drop in the system is upper bounded by the probability that the channel errors in a busy period exceeds value V. Formally, we obtain the following result.

Proposition 2: If the total channel errors in a busy period are bounded above by V with probability p, then p is an upper bound on the probability that the delay vector $\vec{D}$ is guaranteeable. Thus p is an upper bound on the probability for zero packet drop in the system.

In general, probability p depends on the statistics of arrival process, channel characteristics and a scheduling policy used. In our approach, we fix V as a system parameter and ensure that the total number of actual channel errors in a busy period are less than or equal to V with high probability. Let $\mathcal{C}$ indicate a set of active sessions and Z denote the length of the busy period. Now, from the Large Deviation Theory, the convergence of empirical distribution to the steady state distribution is exponential for finite ergodic MC's [16]. Hence for large Z, the channel error rate for session i is equal to δ_i with high probability. Thus, the probability that the total number of channel errors exceed $Z \sum_{i \in \mathcal{C}} \delta_i$ is small.

Using the above observations we develop our admission control framework. Pseudo code for the algorithm is given in Figure 2. Broadly, this algorithm works in the following two steps. In STEP 1, the algorithm obtains an admission control criteria for a given scheduling policy assuming perfect channel condition for all sessions, i.e., none of the sessions see channel errors. Such condition can be obtained using powerful tools like Network Calculus [17], [18]. STEP 2 of the algorithm ensures that the estimate of the total channel errors in a busy period does not exceed the value V. Further, it uses the condition obtained in STEP 1 to verify that the required delay minus V is guaranteeable under the perfect channel.

```
Procedure Admission_Control()
begin
  STEP 1: Obtain admission control condition for a given scheduling policy π
          under perfect channel assumption
  STEP 2:
  When a new session i arrives do the following:
  if (D_i < V) then
     /*  if the required delay is less than V */
     Block the session i;
  else
     Calculate new busy period length using
           Z' = min_{u≥0} {γ[0, u] ≤ (u − V)} ;
     /* γ[0, u] is a maximum data that can arrive in the interval [0, u] */
     Obtain new value of channel errors V' using V' = Z' Σ_{j∈C∪{i}} δ_j ;
     if (V' > V) then
        /* the value of channel errors if i were admitted (V') is greater than a maximum
        allowed value (V) */
        Block the session i;
     else
        D'_i = D_i − V;
        Check guaranteeability of D'_i using condition obtained in STEP 1;
        if D'_i is not guaranteeable with the perfect channel then
           Block the session i;
        else
           Admit session i;
           C = C ∪ {i};
           Update control path state;
end
```

Fig. 2. Pseudo code of a general admission control algorithm for an error-prone wireless channel

IV. SCHEDULING INTERFACE

In this section, we design a scheduling interface to provide packet drop guarantees to individual sessions. To achieve this, we design a scheduling interface that provides *isolation* in a sense that the excessively bad channel for one session does not affect packet drop performance of other sessions.

Recall that in the previous section, we have shown that the proposed admission control algorithm ensures that if the actual number of channel errors for every session does not exceed the estimate $Z\delta_i$, then there is no packet drop in the system. We refer to quantity $Z\delta_i$ as *allowed compensation*. The idea is to design the scheduling interface that provides the allowed compensation to the sessions and penalize them (if required) in case of excessive channel errors. Enforcing penalty allows us to provide isolation and bound the packet drop for every session.

The basic idea behind the scheduling interface is as follows. For every session, interface keeps a count of *compensation slots* in the current busy period. Compensation slots for a session are the slots in which it is chosen for the service and it has a bad channel. The compensation slots represent the amount of compensation provided to the session. If the number of compensation slots is less than $Z\delta_i$, then the scheduler provides additional service to the session in future. If the number is greater than $Z\delta_i$, then the corresponding packets are put in the compensation queue for the session. The packets in the compensation queue are served only when all the primary packets are served. Pseudo code for the scheduling interface is presented in Figure 3.

The proposed interface can be used to upper bound the packet drops experienced by every session as follows.

Proposition 3: The number of packets dropped in a busy period for session i is bounded above by the difference between the number of channel errors in a busy period and the allowed compensation $Z\delta_i$.

```
Procedure Scheduling_Interface()
begin
  When a new time slot t begins
  Assign finish times to packets arrived in the slot;
  Update following sets:
  C_busy = {j : prim_queue(j) is not empty};
  C'_busy = {j : comp_queue(j) is not empty};
  Update ft(j) = finish time of HoL packet in prim_queue(j) ∀j ∈ C_busy;
  Update ft'(j) = finish time of HoL packet in comp_queue(j) ∀j ∈ C'_busy;
  if (C_busy ≠ φ) then
    /* If some session has a packet in primary queue */
    Choose session i = arg min_{j∈C_busy} {ft(j)};
    if (i ∈ C'_busy) then
      /* session i has packets in compensation queue */
      if (Transmit_comp_packet(i) = FALSE) then
        /* Packet is not transmitted from compensation queue */
        Transmit HoL packet of prim_queue(i);
    else
      /* session i's compensation queue is empty */
      Transmit HoL packet of prim_queue(i);
    if (the packet received without errors) then
      /* if i had good channel */
      delete the transmitted packet from the corresponding queue;
      if (Packet is transmitted from compensation queue) then
        Update_comp_counter(i);
    else
      /* if i had a bad channel */
      Update_comp_counter(i);
  else
    flag = FALSE;
    while ((C'_busy ≠ φ) OR (flag = FALSE)) do
      Choose session i = arg min_{j∈C_g ∩ C'_busy} {ft'(j)};
      flag = Transmit_comp_packet(i);
end

Procedure Update_comp_counter(i)
begin
  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
    Comp_counter(i) = Comp_counter(i) + 1;
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. 3. Pseudo code of a general framework with compensation queues given no knowledge of instantaneous channel state

V. Simulation Results and Discussion

A. Simulation Setup

We consider following model for simulations. We model error-prone wireless channel as an ON-OFF Markov process. Transition probabilities from ON state to OFF state and from OFF state to ON state will be denoted by α and β, respectively. We choose α equal to 0.001 and β equal to 0.1. These channel parameters correspond to Raleigh fading channel of mean fade duration 10 slots and channel being good for 99% of time [19]. So for session i, $\delta_i = \frac{\alpha}{\alpha+\beta}$.

The sessions arrive in a Poisson process at the fixed node. Packet arrivals are leaky bucket constrained, i.e., $\vec{A}_i = (\sigma_i, \rho_i)$. The burstiness σ_i and the long term rate ρ_i are assumed to be uniformly distributed between ranges 0-10 packets and 0-0.1 packets per time unit, respectively. The delay requirement D_i for a session is assumed to be uniformly distributed between 5-100 time units. Further we assume that the number of packets a session needs to transfer is exponentially distributed with mean 75. Since packet size is constant, this will directly indicate the total amount of data that a session needs to transmit. With this setup now we show how different scheduling policies can be embedded in the algorithms and how they perform. We run the simulations for 6×10^6 slots.

We incorporate EDF [14] and PGPS [1] in the framework. For STEP 1 in the admission control algorithm (Figure 2), we use admission control conditions obtained in [15] and [1], respectively. We present simulation results for EDF policy. The results are similar for PGPS policy [2].

B. Discussion on Session Blocking

Session blocking is a cup-shaped function of the parameter value V for all the arrival rates considered (Figure 4(a)). This result is intuitive and can be explained as follows. For small values of V, the system experiences higher blocking as the ability to sustain channel errors is reduced. In this scenario, condition $Z \sum_{i \in \mathcal{C}} \delta_i \leq V$ in the admission control algorithm plays a dominating role. In the other extreme, for higher values of V, the session blocking increases again as the admission control algorithm ensures that the delay $D_i - V$ is guaranteeable under perfect channel (Figure 2).

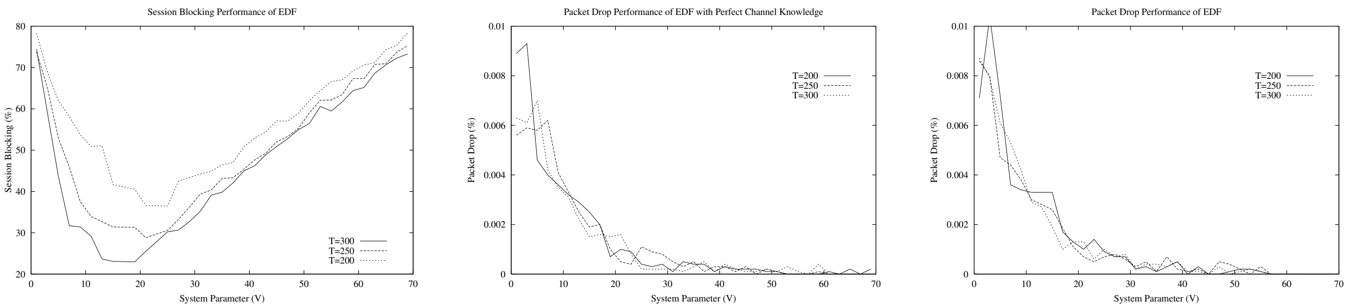
Further, results show that the session blocking increases as the session arrival rate increases. This is because a higher load causes a higher blocking.

C. Discussion on Packet Drop

We first note that because of statistical multiplexing the total number of channel errors are less than or equal to V in most of the busy periods and hence the packet drop in the scheme is low. We observe that the packet drop reduces as value of parameter V increases. This behavior can be explained as follows. Initially (when V takes values between 0 and 25) as value of V increases, the session blocking decreases, hence the busy periods become longer. Hence, the estimate δ_i for the expected number of channel errors becomes accurate. In other words, as busy period becomes large, the probability that the channel errors exceed the average becomes small. Hence the packet drop decreases (Proposition 3).

Further, when the value of V becomes large (greater than 25), the sessions with loose delay constraints are admitted. This implies that the packets can stay in the system for larger amount of time without getting expired. Also, since fewer number of sessions are admitted, system has excess resources, which can be used to serve packets from the compensation queues. Hence the packet drop reduces further.

The above discussion shows that the choice of parameter V is non-trivial and a proper value of V allows us to tune the performance of the system.



(a) (b) (c)

Fig. 4. (a) shows session blocking performance of the EDF policy. (b) and (c) show the packet drop performance of EDF with and without the instantaneous knowledge of channel state at scheduler, respectively. The session arrival rate is $\frac{1}{T}$ in all the figures.

TABLE I
PACKET DROP WITH PLAIN EDF

β_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

TABLE II
PACKET DROP WITH EDF UNDER THE PROPOSED FRAMEWORK

β_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

D. Discussion on Isolation

The proposed framework is designed to provide isolation in the packet drop. Isolation property states that the excessively bad channel for one session does not affect the packet drop for other sessions. We perform the following experiment to verify the isolation property. We consider only two active sessions throughout the duration of simulation. We assume that the estimated channel parameters for both the sessions are α_1 = α_2 = 0.001 and β_1 = β_2 = 0.1. We consider a scenario, where actual channel parameters for session 1 are equal to the estimated value, while actual channel parameters for session 2 deviate from the estimated value. In this scenario, we study the packet drop for both the sessions under EDF augmented as per our scheme (Figure 3) and under the plain EDF scheduling.

Tables I and II provide the experimental results. The column β_2 refers to actual value of channel parameter for session 2. It can be seen that under plain EDF packet drop for both the sessions increases with the increase in deviation of the actual channel parameter value from the estimated value (Table I), while under our framework packet drop increases only for session 2 and it is almost constant for session 1 (Table II). This shows that our framework indeed provides isolation.

REFERENCES

[1] A. K. Parekh and R. G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: The single-node case." *IEEE/ACM Trans. on Networking*, pp. 344–357, Jun. 1993.
[2] P. Chaporkar and S. Sarkar, "Admission control framework to provide guaranteed delay in error-prone wireless channel," University of Pennsylvania, http://www.seas.upenn.edu/∼swati, Tech. Rep., 2002.
[3] V. Bharghavan, S. Lu, and T. Nandagopal, "Fair queueing in wireless networks: Issues and approaches," *IEEE Personal Communications Magazine*, Feb 1999.
[4] S. Lu, V. Bharghavan, and R. Shrikant, "Fair scheduling in wireless packet networks," in *ACM SIGCOMM*, 1997.
[5] S. Lu, T. Nandagopal, and V. Bharghavan, "Design and analysis of an algorithm for fair service in error-prone wireless channels," *Wireless Networks*, pp. 323–343, 2000.
[6] T. E. Ng, I. Stoica, and H. Zhang, "Packet fair queueing algorithms for wireless networks with location dependent errors," Carnegie Mellon University, Tech. Rep. CMU-CS-00-112, 2000.
[7] P. Bhagwat, A. Krishna, and S. Tripathi, "Enhancing throughput over wireless LANs using channel state dependent packet scheduling," in *INFOCOM*, 1996, pp. 1133–1140.
[8] D. A. Eckhardt and P. Steenkiste, "Effort limited fair (ELF) scheduling for wireless networks," in *INFOCOM 2000*, 2000, pp. 1097–1106.
[9] C. Fragouli, V. Shivaraman, and M. Shrivastava, "Controlled multimedia wireless link sharing via enhanced class-based queueing with channel-state dependent packet scheduling," in *INFOCOM*, 1998, pp. 572–580.
[10] J. Gomez, A. T. Campbell, and H. Morikawa, "The Havana framework for supporting application and channel dependent QoS in wireless networks," in *ICNP'99*, 1999, pp. 235–244.
[11] S. Shakkottai and A. L. Stolyar, "Scheduling algorithm for a mixture of real-time and non-real-time data in HDR," in *ITC 2001*, 2001.
[12] P. Ramanathan and P. Agrawal, "Adapting fair queueing algorithms to wireless systems," *Proceedings of MOBICOM*, 1998.
[13] P. Chaporkar and S. Sarkar, "Providing stochastic delay guarantees through channel characteristics based resource reservation in wireless network," in *WOWMOM*, Atlanta, Georgia, 2002.
[14] C. Liu and J. Layland, "Scheduling algorithms for multiprogramming in a hard real time environment," *J. of ACM*, pp. 46–61, Jan. 1973.
[15] L. Georgiadis, R. Guérin, V. Peris, and K. N. Sivarajan, "Efficient network QoS provisioning based on per node traffic shaping," *IEEE/ACM Trans. on Networking*, vol. 4, no. 4, pp. 482–501, Aug. 1996.
[16] A. Shwartz and A. Weiss, *Large Deviations for Performance Analysis: Queues, communications, and computing*. CRC Press, 1995.
[17] R. Agrawal and R. Rajan, "Performance bounds for guaranteed and adaptive services," IBM, Tech. Rep., 1996.
[18] J. L. Boudec and G. Hebuterne, "Application of network calculus to guaranteed service networks," *IEEE/ACM Trans. on Networking*, 1998.
[19] S. Shakkottai and R. Shrikant, "Scheduling real time traffic with deadlines over a wireless channel," *ACM/Beltzer Wireless Networks Journal*, 2001.