

Downlink Scheduling of Heterogeneous Traffic

Aditya Dua and Nicholas Bambos
Department of Electrical Engineering
Stanford University
Stanford, CA 94305
Email: {dua,bambos}@stanford.edu

Abstract—Next generation wireless cellular networks will support a variety of quality-of-service (QoS) sensitive applications like streaming multimedia and high-speed data for downlink users. Channel and QoS aware downlink packet scheduling policies are going to play a key role in efficiently utilizing system resources and enhancing QoS experienced by end users. Schedulers specifically designed to support non-real-time delay tolerant services perform poorly for real-time delay sensitive services, and vice-versa. Scheduler design for supporting a heterogeneous mixture of real-time and non-real-time traffic is a relatively less studied problem. We propose a “quasi-stationary” approach to scheduling of heterogeneous traffic, which involves solving a stationary infinite horizon stochastic shortest-path problem at each scheduling instant. We thoroughly characterize the structural properties of the optimal control associated with the stationary problem, and leverage the intuition thus gained to construct a low-complexity heuristic scheduling policy. We demonstrate the efficacy of the proposed scheduler over benchmark schedulers like the exponential rule via link-level simulations.

Index Terms—Wireless networks, HSDPA, packet deadlines, opportunistic scheduling, resource allocation, dynamic programming.

I. INTRODUCTION

The number of deployed wireless cellular networks continue to grow at a rapid pace. Apart from voice services, next generation cellular networks aim to provide a variety of quality-of-service (QoS) sensitive services like interactive multimedia and high-speed data for downlink subscribers [1]. However, stringent bandwidth and power constraints and the unpredictable nature of the wireless channel render the design of such systems a challenging task.

While second generation (2G) networks vastly benefited from advances in physical layer design principles (coding, constellation design etc.), a *cross-layer design* methodology has attracted much attention in recent years [2]. QoS and channel aware packet scheduling is an important illustration of this design approach. Packet scheduling is going to play a significant role in enhancing the performance of high-speed downlink packet access (HSDPA), a technology introduced in the 3GPP Release 5 specifications to provide high data rates and QoS sensitive services on the downlink [3].

Based on their service requirements, QoS sensitive services can be broadly categorized as *real-time* (RT) and *non-real-time* (NRT). Both these service classes have very different bandwidth and QoS requirements. For instance, video-conferencing, a real-time variable-bit-rate (RT-VBR) application has a delay bound of 40-90 ms and an acceptable loss

rate of 10^{-3} . In contrast, non-real-time applications like web browsing and file transfer have a loss requirement of 10^{-8} but can withstand large delays [4]. Hence arises the need for designing schedulers that explicitly factor in requirements of various classes of services they intend to support.

Scheduler design is guided by the need to optimize one or both of the following objectives: (a) system performance, and (b) user level fairness. Numerous scheduling policies (for both real-time and non-real-time traffic) have been proposed in literature that optimize one of the two objectives or achieve a suitable trade-off between them. See [4]–[9] and references therein.

Scheduler design for supporting a mixture of real-time and non-real-time traffic (heterogeneous traffic) is a relatively less studied problem. The authors in [7] proposed the *exponential rule*, which is throughput-optimal and also provides low average delay. Different weights or priorities are assigned to queues to achieve QoS differentiation. However, the authors do not incorporate packet deadlines into their formulation. The authors in [8] explicitly consider packet deadlines for real-time traffic. However, their choice of objective function to optimize is driven by heuristics and no performance guarantees are provided. To the best of our knowledge, a formal treatment of the scheduling problem for a mixture of real-time and non-real-time traffic which explicitly accounts for packet deadlines does not exist in literature.

In this paper, we provide a first step towards a mathematical treatment of the scheduling problem for heterogeneous traffic. We approximate the scheduling problem by a sequence of stationary infinite horizon stochastic shortest-path problems. We study the structural properties of the optimal control for the stationary problem in a dynamic programming (DP) [11], [12] framework. We also develop a low-complexity approximation to the optimal control, which eliminates the need to explicitly solve the associated Bellman’s equations. With the optimal control and its approximation at our disposal, we propose a *quasi-stationary* approach to scheduling, which involves applying the scheduling decisions of the optimal control for the stationary problem to a “real” system. Our simulation results indicate that the resultant policy is able to provide superlative performance, with very little computational effort.

A. Organization of the paper

In Section II, we formally state the scheduling problem. We construct a simplified stationary model in Section III, and

describe the system dynamics under this model. We study the optimal control for the stationary model and its structural properties in a DP framework in Sections IV and V, and demonstrate the optimality of a switch-over type policy. We develop a quadratic approximation to the optimal switch-over curve in Section VI and propose a heuristic scheduling policy in Section VII. We evaluate the performance of the proposed scheduler via simulations in Section VIII and provide concluding remarks in Section IX.

II. PROBLEM FORMULATION

We consider the downlink of a time-division multiplexed (TDM) wireless communication system. Time is divided into fixed size time-slots. There is one queue corresponding to each downlink user at the base-station (BS). At the beginning of each time-slot, the BS schedules the head-of-line (HOL) packet of a non-empty queue (if any) for transmission to the corresponding downlink receiver or mobile station (MS). The channel for each user exhibits random temporal and spatial fluctuations. While some queues hold real-time packets associated with stringent deadlines, others hold non-real-time packets which seek “best-effort” service in terms of delay performance. An instantaneous and error-free acknowledgment indicating a successful/failed transmission (ACK/NAK) is available to the BS on the uplink. The BS has ideal knowledge of instantaneous downlink channel conditions for all users. Also, the BS is cognizant of deadlines associated with packets in real-time queues and backlogs of non-real-time queues.

Our goal is to design a scheduling policy which offers a “tunable” trade-off between the QoS experienced by the real-time and non-real-time queues. In addition, we would like the scheduler to have desirable properties such as low computational complexity and a parsimonious characterization. Given the multiplicity of problem dimensions and performance criteria, modeling and solving the scheduling problem in its entirety is a mammoth task. Towards this end, we explore mathematical models which capture the essence of the scheduling problem, and yet, remain amenable to analysis.

III. MODEL CONSTRUCTION

A. A “Quasi-Stationary” Model

For most of this paper, we will restrict our attention to a system with two queues: a queue with real-time packets (Q_1) and a queue with non-real-time packets (Q_2). Let us begin by looking at a typical configuration or “snap-shot” of the two-queue system at the beginning of a time-slot. The “snap-shot” comprises of Q_1 (backlogged packets with deadlines), Q_2 (backlogged packets without deadlines), a time-multiplexed transmitter and downlink channels associated with the queues. Next, consider a reduced system based on this “snap-shot”. The reduced model is constructed by discarding all packets other than the HOL packet of Q_1 , and retaining all other constituents of the “snap-shot”. **We will use this reduction as a canonical model for studying scheduler design.**

We will concentrate on studying the optimal scheduling policy for the reduced system. With this optimal policy at

our disposal, we will adopt the following “quasi-stationary” approach to scheduling in a real system:

- (a) Consider the “snap-shot” of the system at the beginning of a scheduling instant.
- (b) Construct a reduction of the “snap-shot”.
- (c) Apply the optimal scheduling decision for the reduction to the original system.

The parameters of the reduced model are updated and the three-step procedure is repeated at each scheduling instant. As we shall see later, the optimal policy for the reduced model is the solution to a stationary infinite-horizon stochastic shortest path problem. Thus, we will approximate the scheduling problem (non-stationary in general) by a sequence of stationary problems, implying the nomenclature “quasi-stationary”. Let us now examine the reduced model more closely.

B. The Real-Time Queue

The HOL packet in Q_1 is characterized by the time-to-expiry (TTE) of its associate deadline, denoted by $n \in \mathbb{N} \cup \{0\}$. This TTE is derived from the differential deadline or *inter-packet deadline* (IPD) of this packet relative to its departed predecessor. For instance, if the receiver expects this packet (call it P) 4 time-slots after the previous packet (call it P') has been received, then the IPD of P relative to P' is 4. If P' departs Q_1 2 time-slots prior to the expiry of its deadline, P inherits a TTE of $4+2=6$ time-slots when it comes to the head of the queue. With such a characterization, it suffices to keep track of *only* the TTE of the HOL packet for making a scheduling decision, rather than the absolute deadlines of all packets in the queue. We also employed this notion in [9], [10] to study scheduling of downlink real-time traffic. In the reduced model, Q_1 stays empty once its HOL packet has departed, either due to successful transmission or due to expiry of its TTE counter. Of course, this is not true for the original system, where the departed packet is replaced by its successor in the queue. If a packet transmission fails, the packet is re-scheduled in a later time-slot, subject to deadline constraints. A cost $\lambda > 0$ is incurred if the HOL packet misses its deadline.

C. The Non-Real Time Queue

The non-real-time queue Q_2 is characterized by its backlog, denoted by $b \in \mathbb{N} \cup \{0\}$. In the reduced model, no new arrivals occur to the queue, so that the backlog eventually reduces to 0. Every packet in this queue incurs a packet holding cost of $c > 0$ for each time-slot it spends in the queue. The BS continues to re-transmit the HOL packet of Q_2 until it is successfully received (stop-and-wait ARQ). Note that packets can arrive arbitrarily to Q_2 in the original system. These arrivals are reflected in the reduced model, since its parameters are updated at each scheduling instant according to the most recent state of the original system.

D. Downlink Wireless Channels

We assume that the channel conditions in the reduced model are time-invariant or static. This static operating point is updated at each scheduling instant in accordance with

the channel variations in the original system. We denote by $s_i \in (0, 1]$ ($i = 1, 2$) the probability of successful reception of a packet from \mathcal{Q}_i , if it is scheduled for transmission. The BS knows only the success probability associated with the downlink channels, and not the actual channel realization (0 or 1), an assumption made in several papers like [6].

E. System Dynamics

Denote by \mathbf{P} the admissible set of all non-idling, non-anticipative and stationary scheduling policies for the reduced model, and let $\mathcal{P}^* \in \mathbf{P}$ be the “optimal” scheduling policy. We will quantify our notion of optimality in the next section. Let us first study the dynamics of the system under a candidate policy \mathcal{P} .

Let us denote the **state** of the system by the two-tuple (n, b) . The states $(0, b)$ and $(n, 0)$ correspond to \mathcal{Q}_1 and \mathcal{Q}_2 being empty respectively, and no scheduling decision needs to be made. In state $(n > 0, b > 0)$, if \mathcal{P} schedules \mathcal{Q}_1 , one of the following state transitions will occur:

- $n > 1$: The transmission is successful with probability s_1 and the new state is $(0, b)$. The transmission fails with probability $(1 - s_1)$ and the new state is $(n - 1, b)$.
- $n = 1$: The new state is always $(0, b)$. The HOL packet of \mathcal{Q}_1 misses its deadline with probability $(1 - s_1)$, and a cost λ is incurred.

If \mathcal{P} schedules \mathcal{Q}_2 , one of the following state transitions will occur:

- $n > 1$: The transmission is successful with probability s_2 and the new state is $(n - 1, b - 1)$. The transmission fails with probability $(1 - s_2)$ and the new state is $(n - 1, b)$.
- $n = 1$: The transmission is successful with probability s_2 and the new state is $(0, b - 1)$. The transmission fails with probability $(1 - s_2)$ and a the new state is $(0, b)$. In both cases, the HOL packet of \mathcal{Q}_1 is dropped and a cost of λ is incurred.

In all cases, a packet holding cost of bc is incurred.

IV. DYNAMIC PROGRAMMING FORMULATION

Having described the system evolution under a typical candidate scheduling policy, we now use a dynamic programming formulation to compute the optimal policy \mathcal{P}^* . Irrespective of the initial state, the system reaches the *absorbing state* $(0, 0)$ with strictly non-zero probability, whereupon no further cost is incurred. Also, the total expected incurred cost is finite¹. Since the parameters of the reduced model are time-invariant, the scheduling problem for the reduced model is a stationary infinite-horizon stochastic shortest path problem, for which an optimal policy \mathcal{P}^* exists. The associated *cost function* $V(n, b)$ satisfies the following *Bellman’s equations*:

$$V(n, b) = \min\{s_1 V(0, b) + (1 - s_1)V(n - 1, b), s_2 V(n - 1, b - 1) + (1 - s_2)V(n - 1, b)\} + bc, \quad (1)$$

¹We can use a dominance argument. Consider an admissible policy \mathcal{P}_1 which always schedules \mathcal{Q}_1 if it is non-empty, and \mathcal{Q}_2 else. Starting in state (n, b) , \mathcal{P}_1 has finite expected cost, upper bounded by $\lambda + bcn + cb(b+1)/2s_2$. The total expected cost for the optimal policy can be no more.

for $n \in \mathbb{N} \setminus \{1\}$, $b \in \mathbb{N}$ and

$$V(1, b) = \min\{s_1 V(0, b) + (1 - s_1)[V(0, b) + \lambda], s_2 V(0, b - 1) + (1 - s_2)V(0, b) + \lambda\} + bc \quad (2)$$

for $n = 1, b \in \mathbb{N}$. To simplify notation, we define for $b, n \in \mathbb{N}$:

$$\begin{aligned} \alpha(n, b) &= V(0, b) - V(n - 1, b) - s_1 \lambda \mathbb{I}_{\{n=1\}} \\ \beta(n, b) &= V(n - 1, b - 1) - V(n - 1, b), \end{aligned} \quad (3)$$

where $\mathbb{I}_{\{n=1\}}$ is an indicator function given by

$$\mathbb{I}_{\{n=1\}} = \begin{cases} 1 & ; \quad n = 1, \\ 0 & ; \quad n > 1. \end{cases}$$

We can now compactly re-write $V(n, b)$ as

$$V(n, b) = \min\{s_1 \alpha(n, b), s_2 \beta(n, b)\} + V(n - 1, b) + bc + \lambda \mathbb{I}_{\{n=1\}}. \quad (4)$$

The behavior of the optimal policy \mathcal{P}^* is completely governed by $\gamma(n, b)$, given by

$$\gamma(n, b) = s_1 \alpha(n, b) - s_2 \beta(n, b). \quad (5)$$

In particular, \mathcal{P}^* schedules \mathcal{Q}_1 in state (n, b) if $\gamma(n, b) \leq 0$, and \mathcal{Q}_2 else. That is, \mathcal{P}^* is fully characterized by the *sign* of $\gamma(n, b)$.

V. STRUCTURAL PROPERTIES OF \mathcal{P}^*

Having established the existence of a stationary optimal policy, we now explore some of its structural properties. We begin with a key property of $\gamma(n, b)$.

Lemma 1: $\gamma(n - 1, b) \leq \gamma(n, b) \leq \gamma(n, b + 1)$, that is, $\gamma(n, b)$ is a non-decreasing function of both its arguments.

Proof: The proof is by mathematical induction. We omit the details due to space constraints. ■

We say a scheduling policy $\mathcal{P} \in \mathbf{P}$ is a *switch-over* policy if there exists a non-increasing switch-over function $\phi : \mathbb{N} \mapsto \mathbb{N} \cup \{0\}$ such that \mathcal{P} schedules \mathcal{Q}_2 if $b > \phi(n)$, and \mathcal{Q}_1 else (adapted from [12]). By virtue of Lemma 1, \mathcal{P}^* is a switch-over policy.

Theorem 1: The optimal packet scheduling policy $\mathcal{P}^* \in \mathbf{P}$ is a switch-over policy.

Proof: \mathcal{P}^* schedules \mathcal{Q}_1 in state (n, b) if $\gamma(n, b) \leq 0$, and \mathcal{Q}_2 else. From Lemma 1, $\gamma(n, b)$ is a non-decreasing function of b . Thus, for fixed $n \in \mathbb{N}$ (say n_0), as b increases, $\gamma(n_0, b)$ changes sign at most once (from negative to positive). If this zero-crossing of $\gamma(n_0, b)$ occurs at $b = \phi(n_0)$, \mathcal{P}^* schedules \mathcal{Q}_1 in state (n_0, b) for $b \leq \phi(n_0)$, and \mathcal{Q}_2 for $b > \phi(n_0)$. Also, from Lemma 1, $\gamma(n, b)$ is a non-decreasing function of n . Thus, for any $n_1 > n_0$ the zero-crossing of $\gamma(n_1, b)$ can occur no later than the zero-crossing of $\gamma(n_0, b)$, as b increases. Equivalently, $\phi(n)$ is a non-increasing function of n . By definition, \mathcal{P}^* is a switch-over policy. ■

Intuitively, for a fixed n , the “backlog pressure” increases as b increases and eventually exceeds the “deadline pressure”.

At this point, the optimal policy *switches* from \mathcal{Q}_1 to \mathcal{Q}_2 . Similarly, increasing n for fixed b releases the “deadline pressure” until it succumbs to the “backlog pressure”, and at this point the optimal policy *switches* from \mathcal{Q}_1 to \mathcal{Q}_2 . A typical switch-over policy is illustrated in Fig. 1.

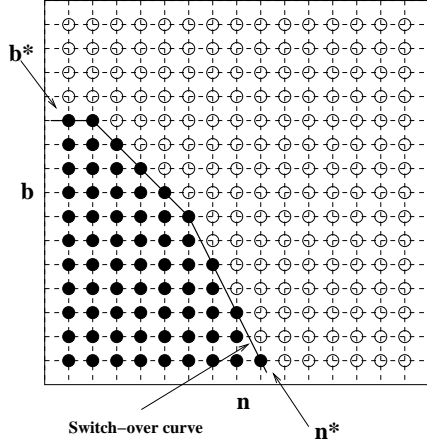


Fig. 1. Illustration of switch-over policy; ● and ○ denote the states (n, b) in which is optimal to schedule \mathcal{Q}_1 and \mathcal{Q}_2 respectively. The extremal points of the switch-over curve, b^* and n^* are also depicted.

Here is a useful property of the cost function V , which we will use in the next section.

Lemma 2:

$$\begin{aligned} (a) \quad V(n, 0) &= (1 - s_1)^n \lambda, \quad \forall n \in \mathbb{N}, \\ (b) \quad V(0, b) &= \frac{cb(b+1)}{2s_2}, \quad \forall b \in \mathbb{N}. \end{aligned}$$

Sketch of Proof: Both properties in Lemma 2 can be proved by solving simple difference equations. In particular,

$$\begin{aligned} V(n, 0) &= (1 - s_1)V(n-1, 0) \\ V(0, b) &= s_2V(0, b-1) + (1 - s_2)V(0, b) + cb, \end{aligned}$$

along with the boundary conditions $V(0, 1) = (1 - s_1)\lambda$ and $V(0, 1) = c/s_2$. ■

VI. APPROXIMATING THE SWITCH-OVER CURVE

Recall that our three-step quasi-stationary approach to scheduling outlined in Section III-A entails solving Bellman’s equations given by (4) to obtain the optimal switch-over curve ϕ at each scheduling instant. From an implementation perspective, this is not an attractive proposition. It is natural to seek approximations to ϕ which can be computed without explicitly solving Bellman’s equations.

Let us denote by $(1, b^*)$ the point of intersection of the optimal switch-over ϕ with the line $n = 1$. Also, let us denote by n^* the largest n for which $\phi(n)$ is strictly positive. Thus, $\phi(n) = 0 \forall n > n^*$. The two extremal points are depicted in Fig. 1. We will construct a quadratic approximation $\hat{\phi}(n)$ of $\phi(n)$, which passes through the points $(1, b^*)$ and $(n^* + 1, 0)$, and is non-increasing. We will explicitly compute b^* and n^* in terms of the problem parameters (λ, c, n, b) .

A. The state $(1, 1)$

\mathcal{P}^* schedules \mathcal{Q}_1 in state $(1, 1)$ if $\gamma(1, 1) \leq 0$, and \mathcal{Q}_2 else. From (3) and (5), $\gamma(1, 1) = s_2V(0, 1) - s_1\lambda$. From Lemma 2, $V(0, 1) = c/s_2$. Thus, \mathcal{P}^* schedules \mathcal{Q}_1 in state $(1, 1)$ if $c \leq s_1\lambda$, and \mathcal{Q}_2 else.

B. Computing b^*

\mathcal{P}^* schedules \mathcal{Q}_1 in state $(1, b)$ if $\gamma(1, b) \leq 0$, and \mathcal{Q}_2 else. From (3) and (5), $\gamma(1, b) = s_2[V(0, b) - V(0, b-1)] - s_1\lambda$. Lemma 2 gives $V(0, b) - V(0, b-1) = bc/s_2$, and hence $\gamma(1, b) = bc - s_1\lambda$. Thus, \mathcal{P}^* schedules \mathcal{Q}_1 in state $(1, b)$ if $c \leq s_1\lambda$, and \mathcal{Q}_2 else. By definition, b^* is the largest b for which \mathcal{P}^* schedules \mathcal{Q}_1 in state $(1, b)$. Since b^* is an integer,

$$\text{we get } b^* = \left\lfloor \frac{s_1\lambda}{c} \right\rfloor.$$

C. Computing n^*

Let us assume that $c \leq s_1\lambda$, that is, \mathcal{P}^* schedules \mathcal{Q}_1 in state $(1, 1)$. If $c > s_1\lambda$, $n^* = 0$. Suppose \mathcal{P}^* schedules \mathcal{Q}_1 in state $(n, 1)$. From Theorem 1, \mathcal{P}^* schedules \mathcal{Q}_1 in all states $(n', 1)$ with $0 < n' < n$. From (4), we have $V(n, 1) = s_1V(0, 1) + (1 - s_1)V(n-1, 1) + c$. Since \mathcal{P}^* schedules \mathcal{Q}_1 in state $(n-1, 1)$, $V(n-1, 1)$ is given by a similar equation in terms of $V(n-2, 1)$. Iteratively,

$$V(n, 1) = (1 - s_1)^{n-1}V(1, 1) + [1 - (1 - s_1)^{n-1}] \left(\frac{c}{s_1} + \frac{c}{s_2} \right).$$

By assumption, \mathcal{P}^* schedules \mathcal{Q}_1 in state $(1, 1)$. Thus, from (4) we have $V(1, 1) = (1 - s_1)\lambda + (1 + 1/s_2)c$. Substitution yields:

$$V(n', 1) = (1 - s_1)^{n'} \left(\lambda - \frac{c}{s_1} \right) + \frac{c}{s_1} + \frac{c}{s_2}, \quad n' \leq n \quad (6)$$

By definition, n^* is the largest n for which $\phi(n) > 0$, or equivalently, \mathcal{P}^* schedules \mathcal{Q}_1 in state $(n, 1)$. Thus, n^* is the largest n for which $\gamma(n, 1) \leq 0$. Recall from (5) that $\gamma(n, 1) = s_1[V(0, 1) - V(n-1, 0)] + (s_2 - s_1)V(n-1, 1)$. We substitute for $V(0, 1)$ and $V(n-1, 0)$ from Lemma 2, set $n' = n^* - 1$ in (6) and finally let $\gamma(n^*, 1) \leq 0$ to get the equivalent condition $(1 - s_1)^{n^*-1} [s_2c + s_1(s_1\lambda - c)] \geq s_2c$. Since n^* is an integer,

$$n^* = \left\lfloor 1 + \frac{\log(s_2) - \log[s_2 + s_1(s_1\lambda/c - 1)]}{\log(1 - s_1)} \right\rfloor.$$

Note that both b^* and n^* are functions of λ/c , that is, they do not depend on the absolute values of λ or c . We will hereafter refer to the quantity λ/c as the *QoS ratio*.

D. Quadratic Approximation

We will use a non-increasing quadratic curve passing through the extreme points $(1, b^*)$ and $(n^* + 1, 0)$ to approximate the switch-over curve ϕ obtained by solving Bellman’s equations. A curve satisfying these conditions is given by

$$\hat{\phi}(n) = b^* \left(1 - \left[\frac{n-1}{n^*} \right]^2 \right), \quad n = 1, \dots, n^*. \quad (7)$$

By definition, $\phi(n) = \hat{\phi}(n) = 0 \forall n > n^*$.

Since we know two points on the optimal switch-over curve, we also explored the idea of a linear approximation, passing through $(1, b^*)$ and $(n^* + 1, 0)$, with slope $-b^*/n^*$. However, a quadratic fit was always seen to be more accurate than a linear fit. We demonstrate the efficacy of the approximation via a numerical example.

Numerical Example 1: Let $s_1 = 0.4, s_2 = 0.6, c = 1, \lambda = 100$ (QoS ratio is 100). Then, $b^* = \lceil s_1 \lambda / c \rceil = 40$ and $n^* = 7$. The quadratic approximation to the switch-curve is given by $\hat{\phi}(n) = 40 \left(1 - \left[\frac{n-1}{7} \right]^2 \right)$. The exact and approximate switch-over curves are depicted in Fig. 2 and are seen to agree closely with each other.

VII. HEURISTIC SCHEDULING POLICY

We have established the optimality of a switch-over policy (\mathcal{P}^*) for a reduced stationary model constructed from a “snapshot” of the original system. In Section VI, we developed a quadratic approximation to the optimal switch-over curve. Also, in Section III-A we outlined a three-step quasi-stationary approach to scheduling which leverages the knowledge of \mathcal{P}^* . We now put all the ingredients together and propose a heuristic scheduling policy for the original system (described in Section II) with one real-time queue (\mathcal{Q}_1) and one non-real-time queue (\mathcal{Q}_2). We denote this policy by Π . At the beginning of a time-slot Π will examine the original system and:

- Set n equal to the TTE of the deadline for the HOL packet of \mathcal{Q}_1 , and b equal to the backlog of \mathcal{Q}_2 .
- Compute s_1 and s_2 from the current downlink channel conditions in the real system. We assume that the BS has available a mapping from downlink SNR to probability of successful transmission for each queue.
- Choose the QoS ratio (λ/c). The QoS ratio could be chosen at the start of a session, and stay fixed thereafter.
- Compute b^* and n^* in terms of s_1, s_2, λ, c .
- Construct the approximate switch-over curve $\hat{\phi}(n)$, as given by (7). Schedule \mathcal{Q}_1 if $b \leq \hat{\phi}(n)$, and \mathcal{Q}_2 else.

The state of the original system is updated based on the scheduling decision of Π . Say, the TTE of the HOL packet of \mathcal{Q}_1 is 6 time-slots and the backlog of \mathcal{Q}_2 is 50 packets at the start of the time-slot. Suppose Π schedules \mathcal{Q}_2 in the current time-slot, and the transmission is successful. Also, 2 new packets arrive to \mathcal{Q}_2 in the current time-slot. Then, for the next time-slot, n^* is set to $6 - 1 = 5$ and b^* is set to $50 - 1 + 2 = 51$. The success probabilities s_1 and s_2 evolve independently of the scheduling decisions.

A. Multiple Queues and Computational Complexity

What happens when the system has $K > 2$ queues? We can construct a reduced model with K queues with a K -dimensional state-space and analyze it in a DP framework. The analysis becomes quite cumbersome due to the explosion in the size of the state-space, and it is also very hard to develop approximations to the optimal control.

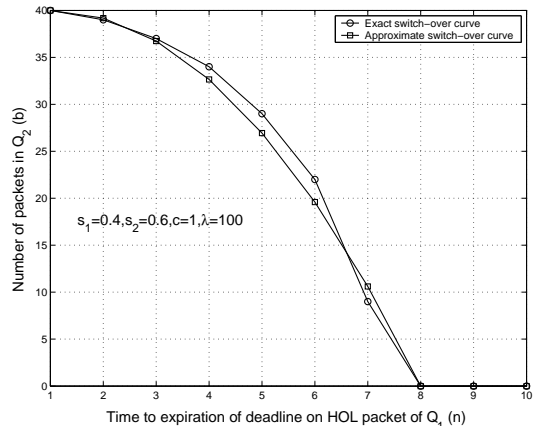


Fig. 2. Exact and approximate switch-over curves for Numerical Example 1

As an alternative, we propose a *pairwise scheduling with knock-out* strategy: Construct a reduced model from a “snapshot” of the original system, as described in Section III-A. Split the K queues into $K/2$ pairs randomly. Consider a pair which comprises of queues \mathcal{Q}_i and \mathcal{Q}_j . If \mathcal{Q}_i is a real-time queue and \mathcal{Q}_j is a non-real-time queue (or vice-versa), use policy Π to choose one of them. If both queues are real-time queues, use the policy proposed in [9], [10] (which is a switch-over policy and permits a piece-wise linear approximation to the optimal switch-over curve) to select one of the queues. If both queues are non-real-time queues, use the well-known $c\mu$ index rule to select one of the queues, with $\mu_i = 1/s_i, \mu_j = 1/s_j^2$. Thus, half the queues get eliminated in each round. The procedure is recursively repeated till exactly one queue survives. The HOL packet of the surviving queue is then scheduled in the original system.

It can be shown that exactly $(K - 1)$ pairwise comparisons are required in the above strategy. Since each pairwise comparison has complexity $O(1)$, the computational complexity of the proposed scheduler is $O(K)$. We established the optimality of the pairwise strategy for real-time queues in [10]. The optimality for non-real-time queues follows easily, since an index rule is used for pairwise comparison. We defer a similar analysis for a system with heterogeneous queues to the journal version of this paper.

VIII. SIMULATION RESULTS

We restrict our attention to a two-queue system and focus on evaluating the performance of scheduling policy Π described in Section VII. Packets with inter-packet deadline $D \in \mathbb{N}$ arrive to the real-time queue \mathcal{Q}_1 at a constant rate $1/D$, while packets arrive to the non-real-time queue \mathcal{Q}_2 according to a Bernoulli process with parameter $p \in (0, 1]$. We use *packet drop rate* and *average delay per packet* as performance metrics for \mathcal{Q}_1 and \mathcal{Q}_2 , respectively. Given the arrival rate p and average backlog in \mathcal{Q}_2 , the average delay is computed from Little’s law [12]. We use the ITU PED-A path profile (3kmph,

²It is easy to establish the optimality of the $c\mu$ rule for choosing between two non-real-time queues in the reduced model

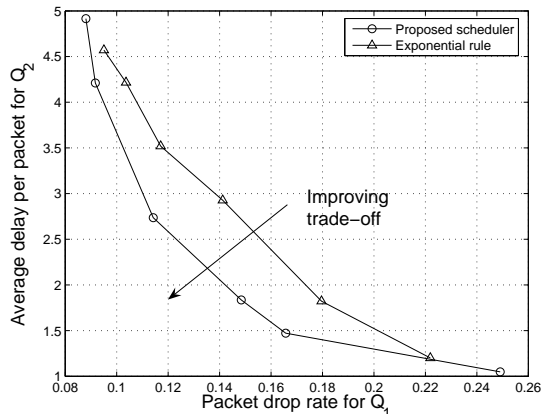


Fig. 3. Average delay per packet for Q_2 v/s packet drop rate for Q_1 . Packets arrive at a constant rate of $1/3$ to Q_1 . Average downlink SNR is same for both queues.

3 independent paths with Rayleigh fading) to simulate the downlink channels, and assume that the mobile receivers accumulate energy ideally from all multi-paths. For the purpose of illustration, we use a simple mapping from the downlink SNR γ to the probability of successful reception s , given by $s(\gamma) = 1 - \exp(-\delta\gamma)$, where $\delta > 0$.

Fig. 3 depicts the trade-off between the QoS experienced by Q_1 and Q_2 under the proposed (II) and exponential rule schedulers. As expected, a reduction in the drop rate of real-time packets results in an increase in average delay experienced by non-real-time packets, and vice-versa. We observe that II offers a better trade-off between the two objectives. The trade-off curve for II is generated by varying the QoS ratio λ/c , while the trade-off curve for the exponential rule scheduler is generated by varying the ratio a_1/a_2 (as defined in [7]).

Fig. 4 depicts the average delay experienced by non-real-time packets in Q_2 when the operating point is chosen such that the real-time packets in Q_1 experience a drop rate of 10%. For a fixed p , the operating point is achieved by appropriately choosing the QoS ratio for II, and the ratio a_1/a_2 for the exponential rule scheduler (via offline computation). We vary p to generate different points on the curve. As expected, the average delay increases as p increases. For fixed p , non-real-time packets scheduled using II experience lower average delay compared to packets scheduled using the exponential rule. The gains are more than 50% in many cases. Alternatively, II offers a higher throughput to Q_2 for a fixed average delay, without affecting the QoS experienced by Q_1 .

IX. CONCLUSIONS

We studied the problem of downlink packet scheduling in wireless cellular networks for supporting a heterogeneous mixture of real-time and non-real-time traffic. We studied a simplified version of the scheduling problem in a dynamic programming framework, for which we established the existence of a stationary optimal control. Based on the structural properties of the optimal control, we proposed a low-

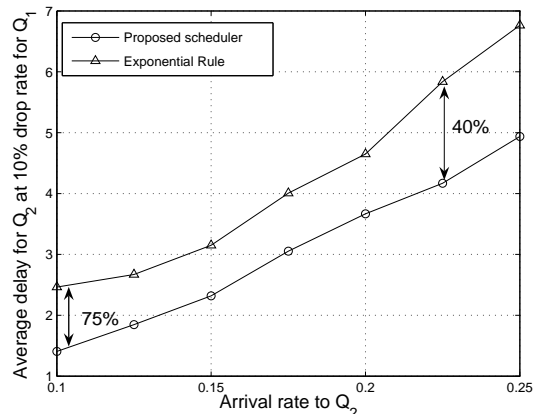


Fig. 4. Average delay per packet v/s packet arrival rate for Q_2 at a packet drop rate of 0.1 for Q_1 . Packets arrive at a constant rate of $1/3$ to Q_1 . Average downlink SNR is same for both queues.

complexity heuristic scheduling policy which can be easily implemented on the downlink of a cellular wireless system. We demonstrated via simulations that the proposed policy outperforms the benchmark exponential rule scheduler. An attractive feature of the proposed scheduler is that it is agnostic to packet arrival and channel statistics. Our ongoing research involves studying optimal scheduler design under more general modeling assumptions, and investigating the interaction of packet scheduling with other resource allocation issues like power and rate control.

REFERENCES

- [1] H. Holma and A. Toskala, *WCDMA for UMTS*, John Wiley and Sons, 3rd. Ed., 2002.
- [2] S. Shakkottai, T. Rappaport and P. Karlsson, "Cross-layer design for wireless networks", *IEEE Communications Magazine*, vol. 41, no. 10, Oct. 2003, pp. 74-80.
- [3] 3GPP, "High speed downlink packet access (HSDPA): overall description", 3GPP, Sophia Antipolis, France, Tech. Spec. 25.308, ver. 5.4.0, Rel. 5, Mar. 2002.
- [4] H. Fattah and C. Leung, "An overview of scheduling algorithms in wireless multimedia networks", *IEEE Wireless Communications*, vol. 9, no. 5, Oct. 2002, pp. 76-83.
- [5] Y. Cao and V.O.K. Li, "Scheduling algorithms in broadband wireless networks", *Proceedings of the IEEE*, vol. 89, no. 1, Jan. 2001, pp. 76-87.
- [6] S. Shakkottai and R. Srikant, "Scheduling real-time traffic with deadlines over a wireless channel", *Wireless Networks*, vol. 8, no. 1, Jan. 2002, pp. 13-26.
- [7] S. Shakkottai and A. Stolyar, "Scheduling algorithms for a mixture of real-time and non-real-time data", *17th International Teletraffic Congress (ITC-17)*, Salvador da Bahia, Sept. 2001.
- [8] K.B. Johansson and D.C. Cox, "An adaptive cross-layer scheduler for improved QoS support of multi-class data services on wireless systems", *IEEE Journal on Selected Areas in Communication*, vol. 23, no. 2, Feb. 2005, pp. 334-343.
- [9] A. Dua and N. Bambos, "Deadline constrained packet scheduling for wireless networks", *62nd IEEE VTC Fall 2005*, Dallas, TX, Sept. 2005.
- [10] A. Dua and N. Bambos, "Wireless packet scheduling with deadlines", *Netlab Technical Report*, SU-Netlab-2005-06/01, Engineering Library, Stanford University.
- [11] D. Bertsekas, *Dynamic Programming and Optimal Control*, vol. 1 and 2, 2nd. Ed., Athena Scientific, 2000.
- [12] J. Walrand, *An Introduction to Queuing Networks*, Prentice Hall, Englewood Cliffs, NJ, 1988.