Measurement-Based Scheduler for Multi-Class QoE Optimization in Wireless Networks

Arjun Anand and Gustavo de Veciana
Department of Electrical and Computer Engineering
The University of Texas at Austin

***Abstract*—Traditional wireless schedulers have been driven by rate-based criteria, e.g., utility maximizing/proportionally fair, and/or queue-based packet schedulers which do not directly reflect the Quality of Experience (QoE) associated with flow-based transactions and services. This paper proposes, a Measurement-Based Delay Optimal (MBDO) scheduler, which optimizes a cost function of the mean flow delays in a multi-class system, e.g,. web interactive, file downloads, etc. In this context the cost function expresses desired trade-offs amongst traffic classes reflecting heterogeneous QoE sensitivities which are nonlinear in the flow delays and/or system loads. To achieve optimality, MBDO scheduling uses measured system variables and knowledge (or measurement) of class flow-size distributions to adapt a weighted Gittins index scheduler. We show that under mild assumptions, and in a stationary regime, that MBDO scheduling is indeed asymptotically optimal. Perhaps more importantly, MBDO schedulers can self-optimize by adapting to slowly varying traffic loads, mixes and flow size distributions. Our extensive simulations confirm the effectiveness at realizing trade-offs and performance of the proposed approach.**

I. Introduction

Next generation wireless networks will likely support an evermore heterogeneous collection of applications ranging across mobile broadband, media, and machine-to-machine type communications. The allocation of Base Station (BS) resources among heterogeneous classes of service with possibly diverse Quality of Experience (QoE) metrics remains a challenging and central problem in wireless system design and is the focus of this paper.

Traditional wireless schedulers have been driven by rate-based criteria, e.g., utility maximization or proportionally fair allocations, which balance the average[1] rates allocated to users and/or queue-based schedulers, which monitor packet queue lengths and/or waiting times. In particular the utility of user/application i is represented via a function $u_i(\cdot)$ of the user's average rate r_i. In the simplest and stationary instance of this framework, scheduling is performed so as to solve the following optimization problem:

$$\max_{\mathbf{r}} \ \{ \ \sum_{i=1}^{n} u_i(r_i) \mid \mathbf{r} \in \mathcal{R}\}, \tag{1}$$

where n is the number of active users, $\mathbf{r} = (r_1, r_2, \ldots, r_n)^T$ and $\mathcal{R}$ is the achievable rate region. In this setting one often assumes users always have data to transmit, i.e., so called *full buffer* model, see e.g. [1]–[3]. This approach clearly does not capture the dynamic nature of transaction/flow based traffic wherein the number of active users changes over time, and wherein QoE is driven by flow-based performance metrics and only indirectly associated with mean rate and/or packet-level delays.

Specifically we shall refer to a flow as the basic data unit whose reception drives the user perceived QoE. In particular, for interactive web browsing a flow could be the content of a web page a user requested, or in the case of a file download the associated with reception of the file. In the context of modern stored video streaming, the video is partitioned into a sequence of small files (video segment) each of which might be considered a flow that should arrive in a timely manner. Several studies have shown that users perceived QoE should be modeled as a non-linear function of the *flow-level delay*, see e.g., [4], [5]. This non-linearity gives us more flexibility in scheduling users' data. For example, for web browsing, it has been shown that users do not perceive any degradation in QoE if the flow delay is less than a certain threshold [4]. So, depending on the system loads, one may not need to be aggressive in allocating resources to web browsing users, possibly to the benefit of others.

In this paper we consider a stochastic model where flows arrive to the system, each with a service requirement in terms of the total amount of bits to be transmitted and they depart after they have been served. We shall assume that there are C classes of users corresponding to different application or service types. Flows arrive at a rate λ_c for class c and we let d_c denote the mean delay experienced by class c flows. We model the end user's QoE through a cost which is an increasing convex function of mean flow delay. The lower the cost, the better the user's QoE. The cost function may depend on the application type allowing one to capture different user/application QoE sensitivities to mean flow delays. The cost function of class c will be denoted by $f_c(\cdot)$. By contrast with rate-based scheduling, we will consider the design of a scheduling policy which solves the following optimization problem:

$$\mathcal{OP}_1: \qquad \inf_{\mathbf{d}^\pi} \ \{ \ \sum_{c=1}^{C} \lambda_c f_c(d_c) \mid \mathbf{d}^\pi \in \mathcal{D} \ \} \tag{2}$$

where $\mathbf{d}^\pi := (d_1^\pi, d_2^\pi, \ldots, d_C^\pi)^T$ is the mean delay vector

This work is supported by Futurewei Technologies

[1]Averages may be computed in an exponentially weighted or moving window ways and thus on different time scales.

realized by policy π and $\mathcal{D}$ is the set of achievable mean delay vectors by all finite mean delay work conserving policies. Note that a work conserving policy need not in general have finite mean delay vector[2]. In $\mathcal{OP}_1$, we scale the cost function of a class with its arrival rate. This is a natural way to represent performance in a dynamic system where one should capture not only high costs, but the number of flows that experience high costs.

In addition to addressing drawbacks associated with the conventional approaches, our model also addresses the need to capture and realize trade-offs in how resources are allocated amongst classes. Our premise is that network operators will want to make QoE trade-offs among applications and that these may be different depending on the system loads. In other words, one should consider optimizing resource allocation for systems not only for heavy loads where such trade-offs are critical, but also for moderate to light loads. As mentioned earlier the trade-offs to be realized can be quite different depending on the load and mix of traffic the system is supporting. For example, when the system is congested, it might be better to give more resources to interactive applications vs large file downloads, so that delay sensitive applications are given priority. However, for lightly loaded systems, allocating more resources to interactive applications will improve their QoE only marginally, once the mean delay is less than a threshold. Therefore, spare resources can be allocated to large file downloads.

In our framework, trade-offs are captured by specifying cost functions for each application. The delay sensitive applications have 'steeper' cost functions after the tolerable delay, as compared to delay tolerant applications. In general, as the system load increases, the mean delays seen by all classes of traffic increase. However, the delay sensitive applications get higher priority because of their steeper cost functions. Therefore, for a range of system loads, a solution to $\mathcal{OP}_1$ will achieve the necessary trade-offs. Next we discuss the related work in flow-level scheduling.

A. Related Work

Flow-level scheduling has been extensively studied in the literature, see [4], [6]–[13]. Some of the works focus only on stability of the system and do not consider delay metrics, see [6], [7]. Several other works target minimization of mean flow delay [10]–[13]. However, as mentioned earlier the users' QoE may not be a linear function of mean delays.

The works most closely related to our work are [4], [8], and [14]. In [4], the authors show that the problem of QoE optimization in wireless networks can be modeled as a Linear Programming problem. However, solving the LP is computationally expensive. Therefore, they develop a heuristic which works well. This paper does not provide any analytical performance results for the heuristic. In [8], the authors develop scheduling policies to satisfy delay based deadlines for various applications. Using simple policies, they achieve the minimum possible deadline violation probability in systems with large amounts of resources (bandwidth and time). The cost functions which we use in our approach can be used to approximate the deadlines and give us more flexibility in allocating resources. In [14], the authors consider an approach which uses cost functions based on delay, however, their work is restricted to only non-pre-emptive scheduling. To the best of our knowledge, this is the first work which considers the minimization of cost functions of the mean delay for general flow size distributions while considering both pre-emptive and non-pre-emptive policies. However, we assume the knowledge (perhaps measured) of flow size distributions which is not assumed in [8], [14]. We deem this a strength since in principle our approach can capture measurable and base station specific characteristics of the offered loads.

Several works such as [6], [10], [11] consider wireless channel models with fast fading. Such a channel model invites the use of opportunistic scheduling policies based on the instantaneous channel conditions. However, in this paper we focus a time invariant channel model. This model is justified when the users are relatively stationary as compared to the time scale of flow dynamics and/or when there is a *channel hardening* effect. Channel hardening occurs when many diverse paths between transmitter and receiver diminishes the effect of fast fading, see [15]. In the sequel we will however incorporate heterogeneous channel strengths as seen by users that have very different channel characteristics due to their different locations, e.g., far or close by, relative to a base station.

B. Our Contributions

In this paper we introduce a Measurement-based Delay Optimal (MBDO) scheduler which minimizes a non-linear cost function of the mean delays experienced in a multi-class system. Starting from a fairly general multi-class M/GI/1 queuing model for a base station we make the following contributions.

1) Extension of Gittins index scheduler: We propose and show a simple extension to the results in [16]. In particular, we show that a weighted Gittins index scheduler (**w**-GITTINSSCHEDULER) will minimize a weighted linear combination of mean delays in a multi-class system. This **w**-GITTINSSCHEDULER scheduler, serves as the workhorse for our MBDO scheduler.

2) MBDO scheduling: We propose the MBDO scheduler which based on system measurements adapts to the system characteristics so as to eventually optimize system performance. In particular, at the end of each queue busy cycle, the MBDO scheduler adapts the weights for a **w**-GITTINSSCHEDULER based scheduler based on measurements to date. Such measurements allow the scheduler to learn the loads on the system, and possibly also to the flow size statistics and optimize scheduling decisions to the specific load and mix the base station is supporting. MBDO scheduler can thus track slow variations in traffic characteristics which

[2]If the service time distribution has a finite mean but infinite second moment, then an M/GI/1 queue served according to a non-preemptive work conserving discipline has an infinite mean delay

might change on the time-scales of few hours in wireless networks, see [17]. The scheduler can in principle also track slow variations in flow size distributions, however, in this paper we assume the knowledge of flow size distributions. *3) Optimality results :* Under mild assumptions on flow size distributions and the knowledge of the minimum of the fraction of total traffic that might arrive to a class, we show that the mean delay vector achieved by our MBDO scheduler converges to the optimal solution of $\mathcal{OP}_1$ in probability.

Overall this approach is quite novel. We are not aware of any proposed measurement-based wireless scheduler able to optimize flow-level delays/trade-offs for a multi-class system. In addition the possibility of tuning scheduling to the traffic characteristics, e.g., flow-size distributions, which may depend on usage patterns in given locations (e.g,, university vs financial district), is novel and intriguing.

C. Organization

This paper is organized as follows. In Section II, we present a simple M/GI/1 queuing model where all flows are served at unit rate. In Section III, we explain about MBDO in detail and prove the asymptotic optimality of our proposed scheme. In Section IV, we extend our scheme for a wireless BS, where different users could have different channel rates. Performance evaluation through simulations is given in Section V.

Notation: In the sequel we denote vectors by bold faced letters and random variables by capital letters. All vectors are column vectors of length C, the number of classes in the system. The components of vectors are represented by normal faced letters, for example, $\mathbf{D}$ denotes a random vector given by $(D_1, D_2, \ldots, D_C)^T$, where T is the transpose operator. Continuous time random processes are written as a function of time, for example, $\{\mathbf{D}(t),\, t \geq 0\}$ is a continuous time vector-valued random process. Discrete time random processes are indexed as follows $\left\{\mathbf{D}^{(k)},\, k \in \mathbb{N}\right\}$. The expectation operator is denoted by $\mathbb{E}[\cdot]$ and the probability of an event A is given by $P(A)$.

II. System Model

Throughout this paper we will develop our scheduler based on a basic multi-class M/GI/1 queuing model, but expect it to be robust to the underlying assumptions. Poisson arrivals are a reasonable model for flow-based transactions and even interactive, i.e, on-off type web browsing, when viewed as an aggregate of reasonably large population. The flow service requirements are generally distributed and again it is reasonable to assume independence amongst flows. We assume that the system supports C classes of flows. Flows of class c arrive as a Poisson process of rate λ_c. The flow sizes are modeled as random variables which are i.i.d. for each class and independent of the flow sizes of other classes. Flow sizes for class c have a distribution function $G_c(\cdot)$ with a mean value of m_c bits. The scheduler does not have prior knowledge of the size of individual flows, however, it does have knowledge of the size distributions, and of the cumulative service each flow has received. Initially we assume that all flows are served at unit rate by the server, thus the stability of queue is assured if $\rho := \sum_{c=1}^{C} \lambda_c m_c < 1$. This will subsequently be relaxed in Section IV.

As mentioned in the introduction we associate a cost function $f_c(.)$ to each class c, which depends on mean flow-delay d_c experienced by flows in that class. We assume that f_c is strictly convex, continuous, and differentiable. Also, f_c is non-decreasing and bounded from below. Let d_c^π be the mean delay of class c under a scheduling policy π. The overall mean delay vector for policy π is denoted by $\mathbf{d}^\pi = (d_1^\pi, d_2^\pi, \ldots, d_C^\pi)^T$. Let $\mathcal{D}$ be the set of mean delay vectors that can be achieved by finite mean delay work conserving policies. We call this as the set of *feasible mean delays*.

III. Cost Minimization

We are interested in finding a scheduling policy π such that $\mathbf{d}^\pi$ solves the following optimization problem.

$$\mathcal{OP}_1: \quad \inf_{\mathbf{d}} \; \{\; f(\mathbf{d}) := \sum_{c=1}^{C} \lambda_c f_c(d_c) \mid \mathbf{d} \in \mathcal{D} \;\}. \tag{3}$$

Note we will show that there is indeed a policy which achieves the infimum.

To solve $\mathcal{OP}_1$, we first consider the following optimization problem:

$$\mathcal{OP}_2 : \inf_{\mathbf{d}} \; \{\; \sum_{c=1}^{C} \lambda_c w_c d_c \mid \mathbf{d} \in \mathcal{D} \;\}, \tag{4}$$

where the weights $w_c, c = 1, 2, \ldots C$ are positive real numbers.

The following corollary, which is a natural consequence of Theorem 5.6 in [16] shows that a Gittins' index based scheduler optimizes $\mathcal{OP}_2$. Below we state the result and then detail the characteristics of such schedulers.

Corollary 3.1: A (**w**-GittinsScheduler) achieves the optimal delays for $\mathcal{OP}_2$. In such a scheduler the Gittins index of a flow is simply scaled by its class weight, and at each time instant the flow with the highest weighted index is scheduled for transmission.

Proof of this result is given in the Appendix A. Next introduce **w**-GittinsScheduler and Gittins indices in detail.

w-GittinsScheduler: Let $\mathcal{A}(t)$ be the set of active flows at time t. For each flow $l \in \mathcal{A}(t)$, we associate a positive real number known as the Gittins index, which is a function of the cumulative service the flow has received, in bits. For a flow l, let its Gittins index be denoted by $\mathcal{G}_l(\cdot)$. At each time t, we scale the Gittins index of a flow by its class weight w_c. We shall refer to this as the *weighted Gittins index*. We schedule the flow with the highest weighted Gittins index at all times. If there are two or more flows with the highest weighted Gittins index, we choose one of the flows at random. Note that the **w**-GittinsScheduler with a given weight vector $\mathbf{w}$ is same as the **w**-GittinsScheduler with weight vector $\kappa\mathbf{w}$, where $\kappa > 0$. Only the relative weights across classes matter in **w**-GittinsScheduler. Therefore, in this paper we

will assume that the weights are normalized to one for **w**-GITTINSSCHEDULER. Next we review the Gittins indices for such dynamic systems given in [16], [18].

Gittins index: Consider a flow which has received a bits of cumulative service. Let $G(\cdot)$ and $\overline{G}(\cdot)$ be the cumulative density function (c.d.f.) and complementary c.d.f. of the flow, respectively. For $\Delta \geq 0$, we define the following

$$R(a,\Delta) := \left(\overline{G}(a) - \overline{G}(a+\Delta)\right)/\overline{G}(a),$$
$$C(a,\Delta) := \left(\int_0^{\Delta} \overline{G}(a+t)\,dt\right)/\overline{G}(a),$$
$$J(a,\Delta) := \frac{R(a,\Delta)}{C(a,\Delta)}.$$

Here $R(a,\Delta)$ and $C(a,\Delta)$ correspond to probability that a flow which has received a bits of service will complete, and the expected time the flow would be busy if it were allocated Δ seconds of service. Therefore, $J(a,\Delta)$ is the ratio of expected *reward* to the expected *cost* of allocating Δ seconds to a flow which has received a bits of service. The Gittins index for an active flow in our queuing model as defined in [16] is given by

$$\mathcal{G}(a) = \sup_{\Delta \geq 0} J(a,\Delta) \tag{5}$$

i.e., the best reward/cost trade-off over all time horizons Δ Computing the Gittins index requires knowledge of flow size distribution. In our setting, different classes of traffic may have different flow size distributions depending on the applications types in the network, and how they are grouped together into classes. Such information can in principle be easily collected by monitoring traffic on the network.

Next we discuss an approach to optimize $\mathcal{OP}_1$ based on a **w**-GITTINSSCHEDULER. The following two lemmas show that $\mathcal{OP}_1$ is a convex problem with a unique minimum which can be realized via a **w**-GITTINSSCHEDULER with appropriate weights.

Lemma 3.2: If $\rho < 1$, then the achievable delay region for work conserving finite mean delay policies $\mathcal{D}$ is a non-empty convex set.

Lemma 3.3: There exists an unique minimizer $\mathbf{d}^*$ for the optimization problem $\mathcal{OP}_1$ and it can be achieved by a weighted Gittins index policy with suitable weights.

Proof: Proofs are given in Appendices B and C, respectively. ∎

In the next sub-section, we will describe our policy in detail.

A. Measurement-Based Delay Optimal (MBDO) Scheduler

The idea underlying MBDO scheduling is to learn an optimal weights setting for **w**-GITTINSSCHEDULER such that optimal delays for $\mathcal{OP}_2$ are also optimizing for $\mathcal{OP}_1$.

We shall decompose the system evolution based on its renewal periods, where each period consists of an idle period and a busy cycle. The weights for the **w**-GITTINSSCHEDULER are fixed for each renewal period but adapted at the end of each renewal cycle based on measurements seen to date. This is exhibited in Figure 1.

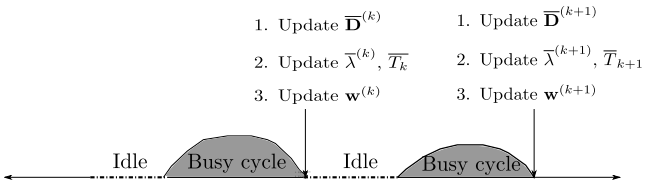

Fig. 1. Sample path of MBDO scheduler

Algorithm 1 Measurement-Based Delay Optimal Scheduling (MBDO)

Initialize: $\overline{\mathbf{D}}^{(0)}, \overline{\boldsymbol{\lambda}}^{(0)}, \overline{T}^{(0)}, \mathbf{w}^{(1)}$ with some non-zero positive values.

Track the following:

- The amount of service given to each flow
- The number of active flows of class c at time t, say $N_c(t)$.

for each renewal cycle k **do**

Run W-GITTINSSCHEDULER($\mathbf{w}^{(k)}$).

$\mathbf{D}^{(k)} \leftarrow$ DELAYESTIMATE($\overline{\boldsymbol{\lambda}}^{(k-1)}, \overline{T}^{(k-1)}, \{N_c(\cdot)\}$)

Updates:

$$\overline{\mathbf{D}}^{(k)} \leftarrow \overline{\mathbf{D}}^{(k-1)} + \epsilon_k\left(\mathbf{D}^{(k)} - \overline{\mathbf{D}}^{(k-1)}\right)$$

$$\mathbf{w}^{(k+1)} = \gamma\left(\overline{\mathbf{D}}^{(k)}\right)\left(\frac{\partial f_1}{\partial d_1}\Big|_{\overline{D}_1^{(k)}}, \frac{\partial f_2}{\partial d_2}\Big|_{\overline{D}_2^{(k)}}, \ldots \frac{\partial f_C}{\partial d_C}\Big|_{\overline{D}_C^{(k)}}\right)^T$$

$$\overline{\boldsymbol{\lambda}}^{(k)} = \frac{(k-1)\,\overline{T}^{(k-1)}}{(k-1)\,\overline{T}^{(k-1)} + T_k}\overline{\boldsymbol{\lambda}}^{(k-1)} + \frac{1}{(k-1)\,\overline{T}^{(k-1)} + T_k}\mathbf{N}^{(k)},$$

$$\overline{T}^{(k)} = \frac{k-1}{k}\overline{T}^{(k-1)} + \frac{1}{k}T_k,$$

end for

procedure DELAYESTIMATE($\overline{\boldsymbol{\lambda}}^{(k-1)}, \overline{T}^{(k-1)}, \{N_c(\cdot)\}$)
return

$$D_c^{(k)} = z\left(\frac{1}{\overline{\lambda}_c^{(k-1)}\overline{T}^{(k-1)}}\right)\int_{k^{\text{th}}\text{ renewal cycle}} N_c(t)\,dt \qquad c = 1, 2, \ldots, C$$

end procedure

procedure W-GITTINSSCHEDULER($\overline{\mathbf{w}}$)
  for Each time slot **do**
    for Each flow in the system **do**
      Compute Gittins index for each flow.
      Scale the Gittins index by its class weight w_c.
    end for
    Schedule the flow with highest w_c.
  end for
end procedure

TABLE I
VARIABLES USED IN MBDO

Name	Description
$\overline{\mathbf{D}}^{(k)}$	Estimate of mean delay upto and including k^{th} renewal cycle.
$\mathbf{D}^{(k)}$	Estimate of mean delay for the policy used in k^{th} renewal cycle.
$\overline{\lambda_c}^{(k)}$	Estimate of mean arrival into class c rate upto and including k^{th} renewal cycle.
$\overline{T}^{(k)}$	Estimate of mean renewal cycle duration upto and including k^{th} renewal cycle.
$N_c(t)$	Number of active flows of class c at time t.
$N_c^{(k)}$	Total number of flows that arrived to class c in k^{th} renewal cycle.
$\mathbf{w}^{(k)}$	Weights used by **w**-GITTINSSCHEDULER in k^{th} renewal cycle.

Pseudo-code for our MBDO scheduler is given in the Algorithm 1 panel. The variables used and their meanings are summarized in Table III-A. The procedure W-GITTINSSCHEDULER simply implements a weighted Gittins index policy during a busy cycle. For simplicity we further divide the time into slots and assume scheduling decisions are made at the beginning of every slot. The slot duration is assumed to be very small as compared to the flow transmission times. The computations performed at the end of a renewal cycle are discussed below.

(1) Delay measurement. We shall estimate the mean delay seen by each class in the k^{th} renewal cycle. it is denoted by $\mathbf{D}^{(k)}$. Further we let $\overline{\mathbf{D}}^{(k)} := \left(\overline{D}_1^{(k)}, \overline{D}_2^{(k)}, \ldots, \overline{D}_C^{(k)}\right)^T$ denote the time-averaged delay vector averaged across renewal cycles up to and including the k^{th} one. Specifically $\overline{\mathbf{D}}^{(k)}$ at the end of k^{th} renewal cycle is updated using the new estimate $\mathbf{D}^{(k)}$ as follows:

$$\overline{\mathbf{D}}^{(k)} \leftarrow \overline{\mathbf{D}}^{(k-1)} + \epsilon_k \left(\mathbf{D}^{(k)} - \overline{\mathbf{D}}^{(k-1)}\right), \tag{6}$$

where $(\epsilon_k \,|\, k \in \mathbb{N})$ is a non-increasing sequence of positive real numbers such that

$$\sum_k \epsilon_k = \infty \quad \text{and} \quad \sum_k \epsilon_k^2 < \infty. \tag{7}$$

For technical reasons, the delay estimate $\mathbf{D}^{(k)}$ is obtained using the procedure DELAYESTIMATE, where the function $z(\cdot)$ is defined as follows:

$$z(x) := \min(x, \lambda/\lambda_{c^*}), \tag{8}$$

where $\lambda_{c^*} = \min\{\lambda_i | i = 1, 2, \ldots, C\}$. The reasoning behind the choice of this estimator is discussed in III-B and we assume the knowledge of the minimum fraction of traffic (λ_{c^*}/λ) that may arrive to any class.

(2) Updating $\overline{\boldsymbol{\lambda}}^{(k)}$ and $\overline{T}^{(k)}$. The estimate of mean arrival rate vector up to and including k^{th} cycle is given by $\overline{\boldsymbol{\lambda}}^{(k)} := \left(\overline{\lambda_1}^{(k)}, \overline{\lambda_2}^{(k)}, \ldots, \overline{\lambda_C}^{(k)}\right)$. Similarly, the estimator for the mean renewal cycle duration up to and including the k^{th} cycle is given $\overline{T}^{(k)}$. They are updated as follows:

$$\overline{\boldsymbol{\lambda}}^{(k)} = \frac{(k-1)\,\overline{T}^{(k-1)}}{(k-1)\,\overline{T}^{(k-1)} + T_k}\overline{\boldsymbol{\lambda}}^{(k-1)} + \frac{1}{(k-1)\,\overline{T}^{(k-1)} + T_k}\mathbf{N}^{(k)}, \tag{9}$$

$$\overline{T}^{(k)} = \frac{k-1}{k}\overline{T}^{(k-1)} + \frac{1}{k}T_k, \tag{10}$$

where T_k is the random variable denoting the length of the k^{th} renewal cycle and $\mathbf{N}^{(k)} := \left(N_1^{(k)}, N_2^{(k)}, \ldots, N_C^{(k)}\right)$ is the random vector denoting the total number of flow arrivals during that cycle for each class.

(3) Adaptation of weights: For the next $(k+1)^{\text{th}}$ renewal cycle, run **w**-GITTINSSCHEDULER with weights given by

$$w_c^{(k+1)} = \gamma\left(\overline{\mathbf{D}}^{(k)}\right)\frac{\partial f_c}{\partial d_c}\Big|_{\overline{D}_c^{(k)}}, \text{ for } c = 1, 2, \ldots, C,$$

where $\gamma\left(\overline{\mathbf{D}}^{(k)}\right)$ is the normalizing factor so that $\mathbf{w}^{(k+1)}$ has unit norm.

The procedure DELAYESTIMATE is crucial to the optimality of MBDO, so we shall discuss it in detail next.

B. Delay Estimates

In our model the duration of renewal periods T_k's are i.i.d. since arrivals are Poisson and service times are i.i.d. Furthermore, the distribution of T_k's is independent of scheduling policy because we are considering only work conserving policies. Note that the delay estimator for class c in the k^{th} renewal period used in the procedure DELAYESTIMATE can be re-written as

$$D_c^{(k)} = U_c^{(k)} + B_c^{(k)} \tag{11}$$

where

$$U_c^{(k)} := \frac{\int_{k^{\text{th}}\text{renewal cycle}} N_c(t)\,dt}{\lambda_c \mathbb{E}[T_k]}, \tag{12}$$

$$B_c^{(k)} := \left[z\left(\frac{1}{\overline{\lambda_c}^{(k-1)}\overline{T}^{(k-1)}}\right) - \frac{1}{\lambda_c \mathbb{E}[T_k]}\right] \tag{13}$$

$$\times \int_{k^{\text{th}}\text{renewal cycle}} N_c(t)dt. \tag{14}$$

In Lemma 3.4, we prove that $U_c^{(k)}$ is an unbiased estimator for mean delay in k^{th} renewal cycle. We also require that this term have finite second moment to prove the convergence results of MBDO. This is proved in Lemma 3.5.

Lemma 3.4: Let $U_c^{(k)} = \frac{\int_{k^{\text{th}}\text{renewal cycle}} N_c(t)dt}{\lambda_c \mathbb{E}[T_k]}$, $c = 1, 2, \ldots, C$, then $U_c^{(k)}$ is an unbiased estimator for the mean delay seen by a typical flow in c^{th} class for the scheduling policy in the k^{th} renewal cycle.

Proof: The proof is given in Appendix D. ∎

Lemma 3.5: Let the fourth moment of flow size distribution for class c be denoted by h_c. If $h_c < \infty$, $c = 1, 2, \ldots, C$, then $\mathbb{E}\left[\left(U_c^{(k)}\right)^2\right] < \infty$, $c = 1, 2, \ldots, C$.

Proof: The proof is given in Appendix E. ∎

The term $B_c^{(k)}$ in (11) represents the bias in the estimator. The function $z(\cdot)$ truncates the value of $1/(\overline{\lambda}^{(k-1)}\overline{T}^{(k-1)})$ to λ/λ_{c^*} which ensures that the term $B_c^{(k)}$ has a finite first moment. We have chosen the value λ/λ_{c^*} for truncation to obtain asymptotic unbiased estimates as indicated in the following lemma.

Lemma 3.6: If $h_c < \infty$ and $z(\cdot)$ is as defined in (8), then $\lim_{k\to\infty} \mathbb{E}\left[\left|B_c^{(k)}\right|\right] \to 0$.

The above lemma shows that $B_c^{(k)}$ converges to zero in expectation. The main idea is to show the almost sure convergence of the sequence $\left(\left|B_c^{(k)}\right| | k \in \mathbb{N}\right)$ to 0 as well as its uniformly integrability. Proof is given in Appendix 3.6. This result is necessary for the convergence result given in the next section.

C. Optimality Results

In this sub-section, we will show the asymptotic optimality of MBDO scheduling and the main result of this paper.

Theorem 3.7: Assuming flow size distributions for all classes have finite fourth moments and $(\epsilon_k \,|\, k \in \mathbb{N})$ satisfies (7), then the MBDO scheduler is such that $\overline{\mathbf{D}}^{(k)}$ converges to $\mathbf{d}^*$ in probability, i.e., for any $\epsilon > 0$

$$\lim_{k\to\infty} P\left(\left|\overline{\mathbf{D}}^{(k)} - \mathbf{d}^*\right| > \epsilon\right) = 0. \tag{15}$$

where $\mathbf{d}^*$ is the unique minimizer of $\mathcal{OP}_1$.

Let us outline the key steps of the proof of this theorem, an leave the details to appendix. Consider the piece-wise constant random process $\overline{\mathbf{D}}(t)$ which is defined as:

$$\overline{\mathbf{D}}(t) = \begin{cases} \overline{\mathbf{D}}^{(k)} & \text{if} \quad t \in \left[\sum_{i=1}^{k-1} \epsilon_i, \sum_{i=1}^{k} \epsilon_i\right) \\ 0 & \text{if} \quad t < 0. \end{cases} \tag{16}$$

The key ideas come from stochastic approximation algorithms wherein as ϵ_k become small, i.e., for large t, the trajectories of the sequence $\overline{\mathbf{D}}(t)$ can be approximated by the trajectories of an associated differential equation for a variable $\overline{\mathbf{x}}(t)$ given by

$$\frac{d\overline{\mathbf{x}}(t)}{dt} = \mathbf{g}^*\left(\overline{\mathbf{x}}(t)\right) - \overline{\mathbf{x}}(t), \tag{17}$$

where $\mathbf{g}^*\left(\overline{\mathbf{x}}\right) := \underset{\mathbf{d}\in\mathcal{D}}{\text{argmin}} \nabla f\left(\overline{\mathbf{x}}\right)^T \mathbf{d}$ and

$$\nabla f(\mathbf{x}) := \left(\lambda_1 \frac{\partial f_1}{\partial d_1}\Big|_{x_1}, \lambda_2 \frac{\partial f_2}{\partial d_2}\Big|_{x_2}, \ldots, \lambda_C \frac{\partial f_C}{\partial d_C}\Big|_{x_C}\right)^T.$$

This can be seen noting that the update equation (6) can be re-arranged as

$$\frac{\overline{\mathbf{D}}^{(k)} - \overline{\mathbf{D}}^{(k-1)}}{\epsilon_k} = \mathbf{D}^{(k)} - \overline{\mathbf{D}}^{(k-1)}. \tag{18}$$

In the above equation, the L.H.S. approximates $\frac{d\overline{\mathbf{D}}(t)}{dt}$ when ϵ_k is small. The term $\mathbf{D}^{(k)}$ should be viewed as an asymptotically unbiased estimate of $\mathbf{g}^*\left(\overline{\mathbf{D}}^{(k-1)}\right)$. Indeed this follows observing that:

1) It follows from Lemmas 3.4 and 3.6, we have shown that $\mathbf{D}^{(k)}$ is an asymptotically unbiased estimate of mean delay under the policy used in k^{th} renewal cycle.
2) Our choice of weights for k^{th} renewal cycle is such that it minimizes $\underset{\mathbf{d}\in\mathcal{D}}{\text{argmin:}} \nabla f\left(\overline{\mathbf{D}}^{(k-1)}\right)^T \mathbf{d}$ in k^{th} renewal cycle.

Therefore, for small ϵ_k we can approximate (18) by (17).

One can then show that the differential equation (17) is globally asymptotically stable and its trajectories converge to the optimal point of $\mathcal{OP}_1$. This follows from the following lemma.

Lemma 3.8: The differential equation given by (17) is globally asymptotically stable and its asymptotically stable point is $\mathbf{d}^*$.

Proof: Proof is given in the Appendix G. ∎

Finally one can use Theorem 2.1 in Chapter 7, [19] to conclude the convergence of $\overline{\mathbf{D}}^{(k)}$ to $\mathbf{d}^*$. The conditions necessary for the theorem are satisfied as a result of Lemmas 3.4, 3.5, 3.6, and 3.8

IV. Modifications to model wireless networks

In the previous section, we considered a multi-class M/GI/1 queue and assumed all flows were served at unit rate. In a wireless network flows destined to different users may experience different service rates due to the heterogeneity in channel conditions they see. For simplicity in this paper we assume (mean) service rates may be heterogenous but are fixed for the duration of a flow. Further modifications can be considered to address opportunism and/or user mobility.

For systems with heterogeneous service rates, one can show that the Gittins index for a flow f need only be scaled by its mean service rate r_f, see [13]. If a flow f has received a cumulative service of x bits and the service rate for the flow is r_f, then its Gittins index $\mathcal{G}_f(\cdot)$ is given by:

$$\mathcal{G}_f(x) = r_f \mathcal{G}(x). \tag{19}$$

where $\mathcal{G}(x)$ is the Gittins index of the flow if it is served at unit rate in a queue. In summary then, the effective weight for a flow would be the product of its class weight w_c and its service rate.

V. Performance Evaluation

In this section, we study the performance of MBDO scheduling through discrete event simulation.

Simulation setup: We consider an M/GI/1 queue with three idealized traffic classes, so as to best understand how MBDO scheduling is performing. We will assume the total service rate is normalized to one bit/second so flow sizes are given in terms of required service time (in seconds). The service rate and the service requirements can be scaled appropriately to study other scenarios too. The three service classes are described in detail below.

1) *Small flows:* Flow sizes for this class are uniformly distributed between 0.1 and 0.3 seconds. The cost function for this class is given by $f_1(d_1) = \frac{1}{2}d_1^2$. This might model web traffic or other interactive applications which are delay sensitive. However, for mean delays upto 1 second, the cost is low.
2) *Medium sized flows:* Flow sizes are uniformly distributed between 0.3 and 0.5 seconds. The cost function used is given by $f_2(d_2) = \frac{1}{3}\left(\frac{d_2}{0.6}\right)^3$, i.e., the delay cost increases steeply after 0.6 seconds. This class represents medium sized flows with tight delay constraints. This could model the segments in a HTTP adaptive video streaming service.
3) *Large files:* Flow sizes have a Pareto distribution with c.c.d.f $\overline{G}(x) = \left(\frac{4}{x+4}\right)^5$, $x \geq 0$. They have a mean service time of 1 second. The Pareto distribution is a heavy-tailed thus this class includes a mix of small and large flows. This could be used to model a variety of file downloads. This class has a cost function $f_3(d_3) = 0.1d_3$. This class is the least sensitive to delay, i.e., most elastic or delay-adaptive.

There are closed form expression for the Gittins index of Uniform and Pareto distributions. For Uniform distribution, the Gittins index is the inverse of the mean residual service time, see [18]. If the flow size is uniformly distributed in the interval $[p, q]$, we have

$$\mathcal{G}_U(a) = \begin{cases} \frac{2}{p+q-2a} & \text{if} \quad 0 \leq a \leq p, \\ \frac{2}{q-a} & \text{if} \quad p < a < q. \end{cases} \tag{20}$$

For Pareto distribution, the Gittins index is equal to its hazard rate, where hazard rate is the ratio of p.d.f. to c.c.d.f., see [18]. Therefore, for our setting, the Gittins index is given by

$$\mathcal{G}_P(a) = \frac{5}{a+4}. \tag{21}$$

In order to see how MBDO realizes trade-offs, we shall fix the arrival rates of two classes and sweep increase that of the third class. Therefore, we study three different cases based on the class for which we sweep the arrival rate. All simulations statistics were obtained based on 4×10^5 flows have been served to completion, giving trends with negligible confidence intervals. In Fig. 2, we have shown the convergence of weights in MBDO for. After about 100 busy cycles, the weights converge. We have used the sequence $\epsilon_k = 1/k, k \geq 1$ to average the delay in MBDO.

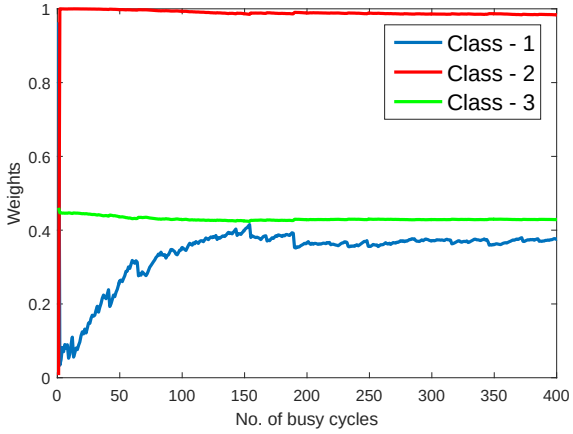

Fig. 2. Weights for all classes as a function of the number of busy cycles for $\lambda_1 = 0.5$, when $\lambda_2 = 1$ and $\lambda_3 = 0.2$ flows/sec.

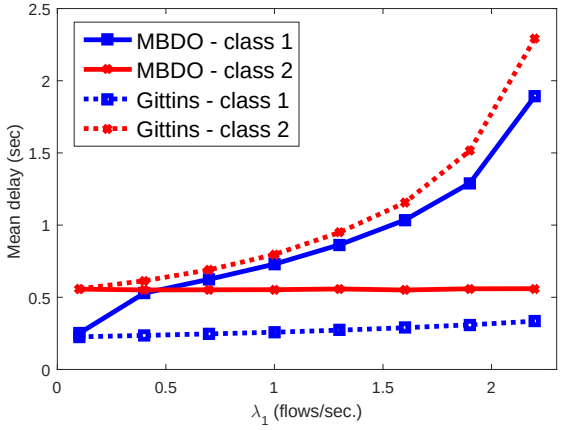

Fig. 3. Mean delays for classes 1 and 2 as a function of λ_1, when $\lambda_2 = 1$ and $\lambda_3 = 0.1$.

We shall mainly compare our scheduler with a mean delay Gittins index scheduler, i.e., **w**-GittinsScheduler with equal weights. The **w**-GittinsScheduler with equal weights is known as Gittins index scheduler in the literature and we will use the same terminology in the sequel. We have also compared our scheduler with the Processor Sharing (PS) scheduler. Note that PS is similar to Proportional Fair when the channels do not change much. However the delay performance for PS (a rate based scheduler) is much worse than MBDO, since it is not geared towards minimizing delays of flow and hence, we cannot illustrate the trade-offs in resource allocation achieved by MBDO when we compare it to PF. The comparison with PS for sweeping the arrival rates of small flows is given in Fig. 4.

1) Sweep arrival rate of small flows: In this scenario, we fix the arrival rates of Classes 2 and 3 (λ_2 and λ_3) at 1 and 0.1 flows/second, respectively. We sweep the arrival rate of Class 1 (λ_1) from 0.1 to 2.2 flows/second. We have plotted the mean delays of Classes 1 and 2 vs λ_1 in Fig. 3. We have not shown the delay performance for the third class in this

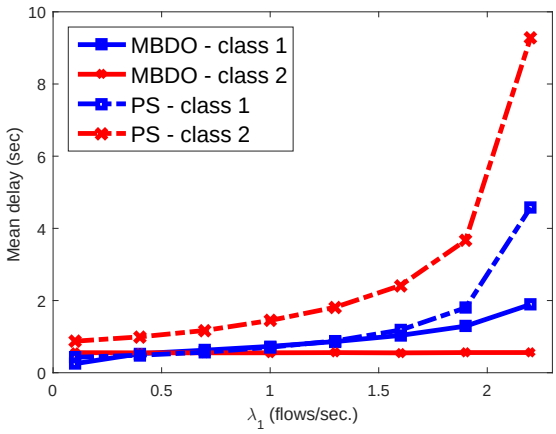

Fig. 4. Mean delays for classes 1 and 2 as a function of λ_1, when $\lambda_2 = 1$ and $\lambda_3 = 0.1$.

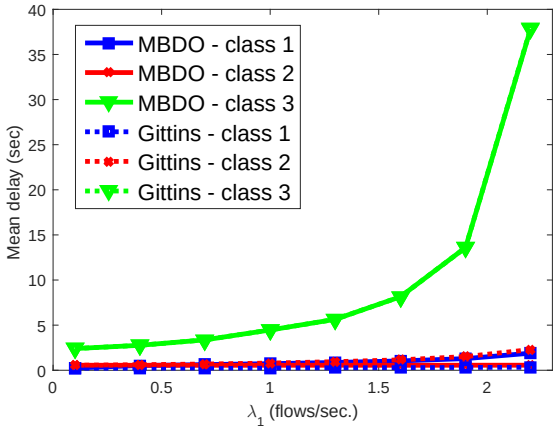

Fig. 5. Mean delays for all classes as a function of λ_1, when $\lambda_2 = 1$ and $\lambda_3 = 0.1$.

Fig.3 because the delay of third class is much larger in both the cases and finer details will be missed. The plot with all three classes is shown in 5. The two key observations obtained from Fig. 3 are as follows:

1) The mean delay for Class 1 flows in MBDO increases much more with λ_1 than for the Gittins index scheduler.
2) The mean delay for Class 2 flows in MBDO stays close to 0.55 sec. even with increasing λ_1, whereas for the Gittins index scheduler it increases by a factor of four on increasing λ_1 from 0.1 to 2.2 flows/sec.

Note that the Gittins index scheduler minimizes the overall mean delay of a typical arrival, i.e., solves $\mathcal{OP}_2$ with equal weights. In other words, by Little's law, it minimizes the mean number of flows in the system. Thus the Gittins index scheduler gives priority to the shorter Class 1 flows at the expense of Class 2 and Class 3 flows. However, for the MBDO scheduler, Class 2 traffic has a very steep cost function after the mean delay of 0.6. As more Class 1 flows arrive into the system, the steep cost function of Class 2 will ensure that the class 2 traffic will get more priority over the Class 1 traffic.

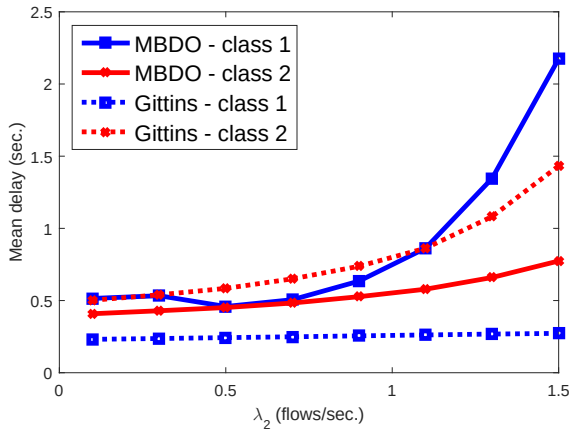

Fig. 6. Mean delays for classes 1 and 2 as a function of λ_2, when $\lambda_1 = 1$ and $\lambda_3 = 0.1$.

Note that the Class 1 traffic can tolerate a mean delay up to 1 sec. without paying too much penalty. Hence, the mean delay of Class 2 does not vary much with λ_1 under MBDO scheduling. Class 3 has lower priority than both Class 1 and 2 as it has the least sensitivity to delay. Therefore, the MBDO is able to protect the most delay sensitive Class 2 traffic from both Class 1 and Class 3 traffic.

2) Sweep arrival rate of medium-sized flows: In this scenario, we keep the arrival rates of Classes 1 and 3 fixed at 1 and 0.1 flows/sec., respectively and the arrival rate of Class 2 is swept from 0.1 to 1.6 flows/sec. We show the mean delays for Classes 1 and 2 vs λ_2 in Fig. 6. An interesting observation is that the mean delay for Class 1 first decreases and then increases on increasing λ_2 in MBDO scheduler. Recall the objective function of $\mathcal{OP}_1$. The overall cost function $f(\cdot)$ increases with λ_2 due to increase in $\lambda_2 f_2(\cdot)$. If the mean delay for Class 2 is less than 0.6 seconds, then value of f_2 does not change much. The only way MBDO can compensate for increasing λ_2 is to decrease the cost of a traffic class, i.e., decrease the mean delay for Class 1. Note that decreasing the mean delay of Class 3 does not help much as it is not so sensitive to delay. Once the mean delay for Class 2 is close to 0.6 seconds, then it dominates the total cost and MBDO stabilizes its delay at the expense of Class 1 traffic. This is in sharp contrast to the Gittins index scheduler which always gives lower mean delays to small flows. Therefore, the mean delay for highly delay sensitive Class 2 traffic is made robust to the changes in its arrival rate in MBDO scheduler.

3) Sweep arrival rate of large flows: Here λ_1 and λ_2 are fixed at 0.5 and 1 flows/sec., respectively, while λ_3 is swept from 0.01 to 0.45 flows/sec. We exhibit the mean delay for classes 1 and 2 vs λ_3 in Fig. 7 and the mean delay for class 3 vs λ_3 in Fig. 8. Note that the mean delays of classes 1 and 2 are not affected by increase in λ_3. There are two reasons for this.

1) For the Pareto distribution, the Gittins index decreases with the cumulative service given to the flow, whereas,

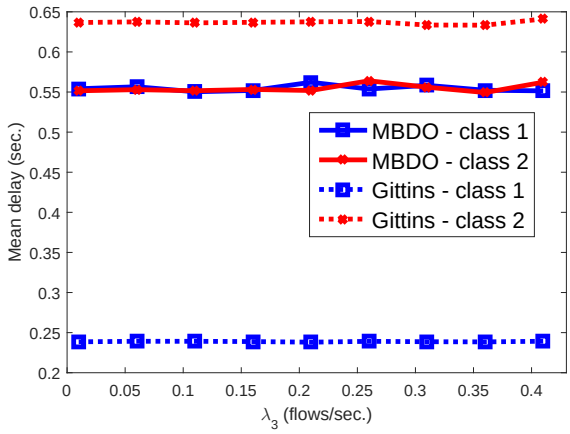

Fig. 7. Mean delays for classes 1 and 32 as a function of λ_3, when $\lambda_1 = 0.5$ and $\lambda_2 = 1$.

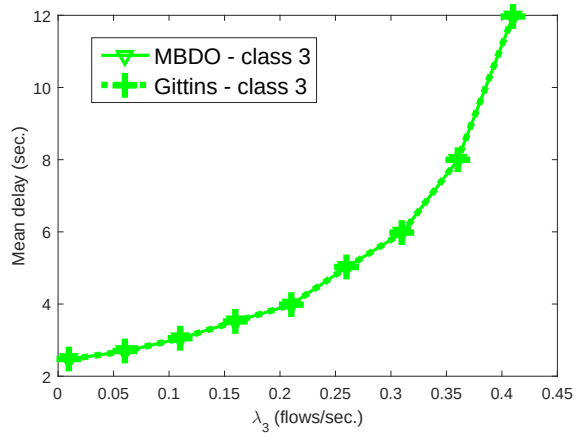

Fig. 8. Mean delays for class 3 as a function of λ_3, when $\lambda_1 = 0.5$ and $\lambda_2 = 1$.

for Uniform distribution it increases. For the parameters which we have used for the Pareto and Uniform distributions, $\mathcal{G}_U(0) > \mathcal{G}_P(0)$. Therefore, Classes 1 and 2 always have higher Gittins indices than class 3. This ensures that Classes 1 and 2 get absolute pre-emptive priority over class 3. Hence, the mean delays of Classes 1 and 2 are not affected by class 3 in Gittins index scheduler.

2) In MBDO, in addition to the Gittins index, we also have the weights associated with the classes. Due to the fact that class 3 is least sensitive to delay, the weights used for classes 1 and 2 are higher than class 3. This along with the characteristics of Gittins indices ensure that classes 1 and 2 get absolute pre-emptive priority over class 3 and hence, they are not affected by class 3.

Due the above mentioned effects, the class 3 always has the least priority in Gittins and MBDO schedulers and therefore, has the same mean delay in both these schedulers. Even though mean delays for classes 1 and 2 are unaffected by class 3 under the Gittins index and MBDO schedulers, they differ in their treatment of Classes 1 and 2. This is because of the effect of cost functions in MBDO which tolerates higher delays for class 1 traffic. Therefore, delay sensitive applications are protected from the changing loads of a delay insensitive application.

VI. Conclusions

In this paper we have proposed a novel delay based approach for QoE optimization in wireless networks. Our proposed scheme MBDO is measurement based and can adapt to slowly varying traffic statistics at the BS. It also achieves optimal trade-offs in resource allocation between application types at various system loads based on their sensitivities to mean delay. Through simulations we have shown that MBDO performs better than mean delay optimal Gittins index and Processor Sharing schedulers.

Appendix

A. Proof of Corollary 3.1

It is shown in Theorem 5.6, [16] that the weighted Gittins index scheduler minimizes the mean expected weighted flow delay in a busy cycle. Using Renewal Reward Theorem (RRT)(see [20]) and the fact that the renewal cycles are identical for all work conserving policies, it can be shown that minimizing the expected weighted flow time in a busy cycle with weights $w_c, c = 1, 2, \ldots, C$ is same as $\mathcal{OP}_2$.

B. Proof of Lemma 3.2

First we will show that the region is convex. Let $\overline{\mathbf{d}}_1$ and $\overline{\mathbf{d}}_2$ be the two mean delay vectors achieved by the two finite mean delay policies π_1 and π_2, respectively. To achieve the mean delay vector $\phi\overline{\mathbf{d}}_1 + (1-\phi)\overline{\mathbf{d}}_2$, $\phi > 0$, use the policy π_1 with probability ϕ and π_2 with probability $1-\phi$, i.i.d. across busy cycles. The set of achievable finite delay vectors is non-empty because the mean delay for a Processor Sharing discipline is finite as long as the first moments of the service times exist and the load ρ is less than one. See [20] for the proof.

C. Proof of Lemma 3.3

For brevity, we define

$$\nabla f(\mathbf{d}) := \left(\lambda_1 \frac{\partial f_1(d_1)}{\partial d_1}, \lambda_2 \frac{\partial f_2(d_2)}{\partial d_2}, \ldots, \lambda_C \frac{\partial f_C(d_C)}{\partial d_C}\right)^T$$

.

In $\mathcal{OP}_1$ we take infimum of a continuous, differentiable, strictly convex, lower bounded, increasing (in all coordinates) function over a convex set $\mathcal{D}$ which is a subset of the positive orthant of $\mathbb{R}^C$. Therefore, the objective function of $\mathcal{OP}_1$ has an infimum and it is uniquely achieved by a vector. Let this infimum achieving vector be denoted by $\overline{\mathbf{d}}^*$. Next we show that there exists a work conserving policy which has its mean delay vector same as $\overline{\mathbf{d}}^*$.

The vector $\overline{\mathbf{d}}^*$ is the optimal solution to $\mathcal{OP}_1$ if and only if the following condition is satisfied.

$$\nabla f\left(\overline{\mathbf{d}}^*\right)^T \left(\overline{\mathbf{d}} - \overline{\mathbf{d}}^*\right) \geq 0, \quad \forall \overline{\mathbf{d}} \in \mathcal{D}. \tag{22}$$

We have to show that the delay vector $\overline{\mathbf{d}}^*$ can be achieved, i.e. it is in $\mathcal{D}$. We have shown in Corollary 3.1 that we can minimize the linear combination of the delays using weighted Gittins index scheduler. Hence, we can minimize $\nabla f\left(\overline{\mathbf{d}}^*\right)^T \overline{\mathbf{d}}$ using weighted Gittins index scheduler using weights as $\nabla f\left(\overline{\mathbf{d}}^*\right)$. From (22), we know that this is a necessary and sufficient condition for optimality. This proves that $\overline{\mathbf{d}}^*$ is in $\mathcal{D}$.

D. Proof of Lemma 3.4

Assume that a given scheduling policy π is used in all busy cycles. Let $N_c(t)$ be the number of flows of class c present in the system at time t. Let N_c be the random variable which denotes the number of customer in class c when the system is stationary. If time instants t_1 and t_2 belong to different busy cycles, then $N_c(t_1)$ is independent of $N_c(t_2)$ because of the independent increment property of Poisson arrivals and the assumption that flow sizes are i.i.d. Therefore we can consider renewal cycles which consist of the idle period and the busy cycle. From the Reward-Renewal theorem, we get that

$$\lim_{\tau\to\infty}\frac{1}{\tau}\int_0^\tau N_i(t)dt = \frac{\mathbb{E}\left[\int_{\text{busy cycle}} N_i(t)dt\right]}{\mathbb{E}\left[T\right]} \quad \text{w.p.1}, \quad (23)$$

where T is a random variable denoting the renewal duration for a typical cycle. Note that the mean renewal cycle duration is same irrespective of the scheduling policy, as long as the policy is work conserving. For stationary queues, we have

$$\lim_{\tau\to\infty}\frac{1}{\tau}\int_0^\tau N_i(t)dt = \mathbb{E}\left[N_i\right] \quad \text{w.p.1.} \quad (24)$$

Using Little's law, we get $\mathbb{E}\left[N_i\right] = \lambda_i\mathbb{E}\left[D_i\right]$, where D_i is the random variable which denotes the stationary mean delay seen by a typical arriving customer. Substituting this in (23), we get that

$$\mathbb{E}\left[D_i\right] = \frac{\mathbb{E}\left[\int_0^T N_i(t)dt\right]}{\lambda_i\mathbb{E}\left[T\right]}. \quad (25)$$

If we define $D_i := \frac{\int_0^T N_i(t)dt}{\lambda_i\mathbb{E}[T]}$, then from the above expression D_i is an unbiased estimator for the delay of class i flows under the policy used in the given renewal cycle.

E. Proof of Lemma 3.5

Let us look at k^{th} busy cycle. Let $N^{(k)}$ be the total number of jobs that arrived in the busy cycle. Then

$$\mathbb{E}\left[\left[\int_{k^{\text{th}}\text{ busy cycle}} N_i(t)dt\right]^2\right] \leq \mathbb{E}\left[\left(N^{(k)}T_k\right)^2\right]. \quad (26)$$

From CauchySchwarz inequality, we get that

$$\mathbb{E}\left[\left(N^{(k)}T_k\right)^2\right] \leq \sqrt{\mathbb{E}\left[\left(N^{(k)}\right)^4\right]\mathbb{E}\left[T_k^4\right]}. \quad (27)$$

Based on the analysis of the distribution of busy cycle duration in Chapter 27, [20], it can be shown that $\mathbb{E}\left[\left(N^{(k)}\right)^4\right]$ and $\mathbb{E}\left[T_k^4\right]$ are finite when $h_c < \infty, c = 1, 2, \ldots, C$ and $\rho < 1$.

F. Proof of Lemma 3.6

From the definition of $\overline{\lambda_c}^{(k)}$ and $\overline{T}^{(k)}$ in (9) it is true that

$$\lim_{k\to\infty}\overline{\lambda_c}^{(k)} = \lambda_c, \quad \text{w.p.1.} \quad (28)$$

$$\lim_{k\to\infty}\overline{T}^{(k)} = \mathbb{E}\left[T\right] \quad \text{w.p.1.} \quad (29)$$

Since the function $1/x$ is continuous when $x > 0$, the above results would ensure that

$$\lim_{k\to\infty}\left|\frac{1}{\overline{\lambda_c}^{(k)}\overline{T}^{(k)}} - \frac{1}{\lambda_c\mathbb{E}\left[T\right]}\right| = 0 \quad \text{w.p.1.}, \quad (30)$$

However, to prove that the above term converges to zero in expectation, we have to show uniform integrability of the sequence $\left\{\left|\frac{1}{\overline{\lambda_c}^{(k)}\overline{T}^{(k)}} - \frac{1}{\lambda_c\mathbb{E}[T]}\right| \mid k \in \mathbb{N}\right\}$. Therefore, we introduce the thresholding function $z\left(\cdot\right)$.

Let us consider a threshold $\theta > 0$ and define $z\left(\cdot\right)$ as follows

$$z\left(x\right) := \min\left(\theta, x\right), \quad x \geq 0. \quad (31)$$

If the value of θ is such that $\theta \geq \max\left\{\frac{1}{\lambda_c\mathbb{E}[T]} \mid c = 1, 2, \ldots, C\right\}$, then we have that

$$\lim_{k\to\infty}\left|z\left(\frac{1}{\overline{\lambda_c}^{(k)}\overline{T}^{(k)}}\right) - \frac{1}{\lambda_c\mathbb{E}\left[T\right]}\right| = 0\,\forall c \quad \text{w.p.1.} \quad (32)$$

This is because we have chosen θ such that $z\left(\frac{1}{\lambda_c\mathbb{E}[T]}\right) = \frac{1}{\lambda_c\mathbb{E}[T]}\forall c$. Since $\left|z\left(\frac{1}{\overline{\lambda_c}^{(k)}\overline{T}^{(k)}}\right) - \frac{1}{\lambda_c\mathbb{E}[T]}\right|$ is bounded for all k, there exists a constant $B < \infty$ such that

$$\mathbb{E}\left[\left|z\left(\frac{1}{\overline{\lambda_c}^{(k)}\overline{T}^{(k)}}\right) - \frac{1}{\lambda_c\mathbb{E}\left[T\right]}\right|^2\right] < B \quad \forall k. \quad (33)$$

This is a sufficient condition for uniform integrability of the sequence.

Next we will show that $\theta = \lambda/\lambda_c^*$ is a good choice for the threshold. We require that $\theta \geq \max\left\{\frac{1}{\lambda_c\mathbb{E}[T]} \mid c = 1, 2, \ldots, C\right\}$. This is ensured if

$$\theta \geq \frac{1}{\lambda_{c^*}\mathbb{E}\left[T\right]}. \quad (34)$$

From [20], we know that

$$\mathbb{E}\left[T\right] = 1/\lambda + \frac{\rho/\lambda}{1-\rho}. \quad (35)$$

Substituting the above expression into the inequality (34), and using the fact that $\frac{\rho/\lambda}{1-\rho} \geq 0$, we get that

$$\theta \geq \lambda/\lambda_{c^*}. \quad (36)$$

G. Proof of Lemma 3.8

A differential equation is globally asymptotically stable if for any initial condition, eventually it converges to the equilibrium point. Here the equilibrium point is the place where $\frac{d\overline{\mathbf{x}}(t)}{dt} = 0$. From (17), the equilibrium point $\mathbf{x}^*$ satisfies:

$$\mathbf{g}^*\left(\mathbf{x}^*\right) = \mathbf{x}^*. \tag{37}$$

From the definition of $\mathbf{g}^*\left(\cdot\right)$, this implies that

$$\nabla f\left(\mathbf{x}^*\right)^T \mathbf{x}^* \leq \nabla f\left(\mathbf{x}^*\right)^T \mathbf{d} \quad \forall\, \mathbf{d} \in \mathcal{D}. \tag{38}$$

From (22), this implies that $\mathbf{x}^*$ is the optimal solution for $\mathcal{OP}_1$, which we proved is unique.

To prove that the differential equation (17) is globally asymptotically stable, it is enough to show that we can construct a Lyapunov function $L\left(\mathbf{d}\right)$ which has a negative drift. Let $L\left(\mathbf{d}\right) := f\left(\mathbf{d}\right) - f\left(\mathbf{x}^*\right)$.

$$\frac{dL\left(\mathbf{d}\right)}{dt} = \frac{df\left(\mathbf{d}\right)}{dt}, \tag{39}$$

$$= \nabla f\left(\mathbf{d}\right)^T \frac{d\mathbf{d}}{dt}, \tag{40}$$

$$= \nabla f\left(\mathbf{d}\right)^T \mathbf{g}^*\left(\mathbf{d}\right) - \nabla f\left(\mathbf{d}\right)^T \mathbf{d}, \tag{41}$$

$$\leq 0. \tag{42}$$

The last inequality follows from the definition of $\mathbf{g}^*\left(\cdot\right)$. However, we have to show a strict negative drift for the Lyapunov function. In the RHS of (41), $\nabla f\left(\mathbf{d}\right) > \mathbf{0}, \forall\, \mathbf{d} \in \mathcal{D}$. This is because of the assumption of strict convex and increasing cost functions. Therefore, for the R.H.S. of (41) to be zero, the only possibility is $\mathbf{g}^*\left(\mathbf{d}\right) = \mathbf{d}$. This happens only at the equilibrium point $\mathbf{x}^*$, which is same as the unique solution to $\mathcal{OP}_1$. Hence, the drift is strictly negative when the delay vector $\mathbf{d}$ is away from the equilibrium point.

REFERENCES

[1] D. Tse and P. Vishwanath, *Fundamentals of Wireless Communications*. Cambridge University Press, 2005.

[2] F. Kelly, A. Maulloo, and D. Tan, "Rate control in communication networks: shadow prices, proportional fairness and stability," in *Journal of the Operational Research Society*, vol. 49, 1998.

[3] K. Seong, M. Mohseni, and J. Cioffi, "Optimal resource allocation for ofdma downlink systems," in *Information Theory, 2006 IEEE International Symposium on*, Jul. 2006, pp. 1394–1398.

[4] M. Proebster, M. Kaschub, T. Werthmann, and S. Valentin, "Context-aware resource allocation for cellular wireless networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2012, no. 1, pp. 1–19, 2012.

[5] P. Ameigeiras, J. J. Ramos-Munoz, J. Navarro-Ortiz, P. Mogensen, and J. M. Lopez-Soler, "Qoe oriented cross-layer design of a resource allocation algorithm in beyond 3g systems," *Comput. Commun.*, vol. 33, no. 5, pp. 571–582, Mar. 2010.

[6] S. Borst, "User-level performance of channel-aware scheduling algorithms in wireless data networks," in *Proc. INFOCOM*, vol. 1, Mar. 2003, pp. 321–331.

[7] S. Borst and M. Jonckheere, "Flow-level stability of channel-aware scheduling algorithms," in *Proc. INFOCOM*, Apr. 2006, pp. 1–6.

[8] H. Wu, X. Lin, X. Liu, and Y. Zhang, "Application-level scheduling with deadline constraints," in *Proc. INFOCOM*, Apr. 2014, pp. 2436–2444.

[9] J. Ghaderi, T. Ji, and R. Srikant, "Connection-level scheduling in wireless networks using only mac-layer information," in *Proc. INFOCOM*, Mar. 2012, pp. 2696–2700.

[10] B. Sadiq and G. de Veciana, "Balancing srpt prioritization vs opportunistic gain in wireless systems with flow dynamics," in *Teletraffic Congress (ITC), 2010 22nd International*, Sep. 2010, pp. 1–8.

[11] S. Aalto, A. Penttinen, P. Lassila, and P. Osti, "Optimal size-based opportunistic scheduler for wireless systems," *Queueing Systems*, vol. 72, no. 1, pp. 5–30, 2012.

[12] S. Aalto and P. Lassila, "Impact of size-based scheduling on flow level performance in wireless downlink data channels," in *Managing Traffic Performance in Converged Networks*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2007, vol. 4516, pp. 1096–1107.

[13] I. Taboada, J. Fajardo, F. Liberal, and B. Blanco, "Size-based and channel-aware scheduling algorithm proposal for mean delay optimization in wireless networks," in *Proc. ICC*, June 2012, pp. 6596–6600.

[14] C.-p. Li and M. J. Neely, "Delay and Power-Optimal Control in Multi-Class Queueing Systems," *ArXiv e-prints*, Jan. 2011.

[15] B. M. Hochwald, T. L. Marzetta, and V. Tarokh, "Multiple-antenna channel hardening and its implications for rate feedback and scheduling," *IEEE Trans. Inf. Theory*, vol. 50, no. 9, pp. 1893–1909, Sept. 2004.

[16] J. Gittins, K. Glazebrook, and R. Weber, *Multi-armed Bandit Allocation Indices*, 2nd ed. Qiley, 2011.

[17] N. E, X. Chu, W. Guo, and J. Zhang, "User data traffic analysis for 3g cellular networks," in *Communications and Networking in China (CHINACOM), 2013 8th International ICST Conference on*, Aug. 2013, pp. 468–472.

[18] S. Aalto and U. Ayesta, "Optimal scheduling of jobs with a dhr tail in the m/g/1 queue," in *Proceedings of the 3rd International Conference on Performance Evaluation Methodologies and Tools*, ser. ValueTools '08, 2008, pp. 50:1–50:8.

[19] H. J. Kushner and G. G. Yin, *Stochastic Approximation Algorithms and Applications*. Springer, 1997.

[20] M. Harchol-Balter, *Performance Modeling and Design of Computer Systems: Queueing Theory in Action*. Cambridge University Press, 2013.