Delay and Rate-Optimal Control in a Multi-Class Priority Queue with Adjustable Service Rates

Chih-ping Li
Laboratory for Information and Decision Systems
Massachusetts Institute of Technology

Michael J. Neely
Department of Electrical Engineering
University of Southern California

***Abstract*—We study two convex optimization problems in a multi-class $M/G/1$ queue with adjustable service rates: minimizing convex functions of the average delay vector, and minimizing average service cost, both subject to per-class delay constraints. Using virtual queue techniques, we solve the two problems with variants of dynamic $c\mu$ rules. These algorithms adaptively choose a strict priority policy, in response to past observed delays in all job classes, in every busy period. Our policies require limited or no statistics of the queue. Their optimal performance is proved by Lyapunov drift analysis and validated through simulations.**

I. Introduction

Dynamic control over multi-class queueing systems has attracted significant attention for decades due to its wide applications in computers, communication networks, and manufacturing systems. One particularly useful tool is to characterize the achievable region of a performance measure of interest, then use optimization methods to develop optimal control policies (see [1], [2] for examples). One celebrated result that holds when the performance region is a polymatroid is that a strict priority policy called the $c\mu$ rule minimizes linear costs (such as sum of average queue lengths) [3]. A followup question is whether such simple policies exist for more complex objectives. In this paper, we show that convex optimization problems in an $M/G/1$ queue, which has a polymatroidal delay region, can have adaptive online policies as simple as the $c\mu$ rule.

Consider an $M/G/1$ queue serving N independent classes of Poisson arrivals. The controller serves jobs one at a time in a non-preemptive fashion. After completing a job it makes a decision about which job class to serve next. It can also dynamically adjust the service rate. Let the arrival rate of class $n \in \{1, \ldots, N\}$ be λ_n, and the random variable S_n (with mean $\mathbb{E}[S_n]$) denote the size of a class n job. All job sizes are independent across classes and independent and identically distributed (i.i.d.) within each class. As a technical detail, we assume that the first four moments of S_n are finite for all n; the distribution of S_n is otherwise arbitrary. When a job arrives, we only know its class but not its actual size. The server is assumed to have an adjustable service rate $\mu(P(t))$, incurring an instantaneous cost $P(t)$ at time t (such as a power cost). Assume $\mu(\cdot)$ is increasing with $\mu(0) = 0$.

Let $\overline{W}_n$ be the average queueing delay vector for class n.[1] We consider the two problems:

1) Minimizing a separable convex function of the average delay vector $(\overline{W}_n)_{n=1}^N$ subject to delay constraints $\overline{W}_n \leq d_n$ for every class $n \in \{1, \ldots, N\}$. Here, we assume a constant service rate. That is, we optimize only over scheduling decisions, and do not consider service rate control.
2) Under dynamic allocations of the service rate, we minimize average service cost subject to delay constraints $\overline{W}_n \leq d_n$ for every class $n \in \{1, \ldots, N\}$.

Solving the first problem enables delay fairness across job classes. This is in the same spirit as the well-known rate proportional fairness [4] or utility proportional fairness [5], and we show in Section III-A that the objective function for *delay proportional fairness* is quadratic (rather than logarithmic). One application of the second problem is to minimize power consumption of a computer using dynamic speed scaling [6], while providing delay guarantees to different traffic streams.

When the service rate is constant, the set of all achievable delay vectors in a nonpreemptive $M/G/1$ queue is a polymatroid, a special polytope with vertices achieved by strict priority policies [7]. It follows that the optimal solution to the first problem above is achieved by a randomized policy that updates priorities over busy periods according to a stationary distribution. We are thus motivated to find an optimal dynamic policy that updates priorities over busy periods. In the second problem with service rate control, we find the optimal policy that updates priorities and service rates over busy periods.

Our construction of dynamic priority policies is based on using virtual queues to monitor, in each job class, the amount of past observed delays violating the delay constraint (stored as virtual queue backlogs). Then, at decision epochs (end of busy periods), job classes with more severe delay violations are offered higher priorities until the next decision epoch, and so on. Technically, using Lyapunov drift analysis, we show

This work was performed when the first author was with the Department of Electrical Engineering in the University of Southern California, Los Angeles, CA.

This material is supported in part by one or more of the following: the DARPA IT-MANET program grant W911NF-07-0028, the NSF Career grant CCF-0747525, NSF grant 0964479, the Network Science Collaborative Technology Alliance sponsored by the U.S. Army Research Laboratory W911NF-09-2-0053.

[1] We only consider *queueing delay*, not *system delay* (queueing plus service), in this paper. In a nonpreemptive setting, average system and queueing delay differ only by the mean service time. Thus, results in this paper can naturally be generalized to optimizing system delays. We use *delay* and *queueing delay* interchangeably in the rest of the paper.

that policies stabilizing the virtual queues are online policies solving the optimization problems in the $M/G/1$ queue. These policies make a "max-weight" decision in every busy period, where the decisions turn out to be a $c\mu$ rule assigning priorities by sorting weighted virtual queue backlogs.[2] We show that the resulting dynamic $c\mu$ rules yield performance that is $O(1/V)$ away from optimal, where $V > 0$ is a control parameter that can be chosen sufficiently large for optimality, with a tradeoff in convergence time [10, Remark 5.11].

In the queueing literature, compared to the vast study of showing strict priority policies (such as the $c\mu$ rule) minimize linear costs, the study of constrained convex optimization problems is relatively limited; see [11]–[14] for examples. Adaptive policies for the control of single-server queues are developed in [14], [15] using stochastic approximation. Our adaptive control method in this paper is conceptually simpler than stochastic approximation, and naturally incorporates time average constraints.

State-dependent allocation of service rates in a single-server queue is addressed in [16], [17] and references therein. The usual approach uses dynamic programming (DP) ideas to show the monotonic structure of optimal policies. DP seems inadequate for our problems because the multi-class assumption leads to a multi-dimensional state space suffering from the curse of dimensionality. In addition, DP seems complicated to incorporate time-average constraints, and requires full statistics of the queue; our policies need only limited statistics. In this paper we focus on policies that update service rates over busy periods; this approach leads to tractable analysis and simplified control.

In the rest of the paper, Section II presents useful notations, definitions, and control policies of interest in this paper. Section III solves the first convex delay optimization problem, and introduces the notion of delay proportional fairness. Section IV solves the second service rate control problem. Simulations are provided in Section V.

Due to space limits, we only present main results in this paper. See [10, Chapter 5] for detailed proofs and the technical construction of the dynamic $c\mu$ rules.

II. Notations, Definitions, and Policies

We consider the $M/G/1$ queue as a frame-based system, where each frame consists of an idle period and the following busy period. Let t_k, $k \in \mathbb{Z}^+$, be the start of the kth frame. The kth frame is $[t_k, t_{k+1})$ with size $T_k \triangleq t_{k+1} - t_k$. Define $t_0 = 0$ and assume the system is initially empty. Let $A_{n,k}$ be the set of class n arrivals in frame k. For each job $i \in A_{n,k}$, let $W_{n,k}^{(i)}$ be its queueing delay.

The average delay under policies that we propose later may not have well-defined limits. Thus, inspired by [18], we define

$$\overline{W}_n \triangleq \limsup_{K\to\infty} \frac{\mathbb{E}\left[\sum_{k=0}^{K-1} \sum_{i\in A_{n,k}} W_{n,k}^{(i)}\right]}{\mathbb{E}\left[\sum_{k=0}^{K-1} |A_{n,k}|\right]} \tag{1}$$

[2]For the second service rate control problem, we need a "ratio max-weight" principle [8], [9] to find the optimal priority assignment in every busy period, because the mean size of a busy period is a function of the service rate.

as the average queueing delay for class $n \in \{1, \ldots, N\}$, where $|A_{n,k}|$ is the number of class n arrivals in frame k. We only consider average delays sampled at frame boundaries for simplicity. See [18] for a discussion on the definition of (1).

The next two assumptions specify the class of control policies considered in this paper.

Assumption 1. We focus on scheduling policies that are work-conserving, non-preemptive, and non-anticipative. The controller can serve one job at a time. At the completion of every service, it makes a dynamic decision about which job class to serve next. Jobs in each class can be served in arbitrary order.

Assumption 2. When the service rate is adjustable, a fixed service rate $\mu(P_k)$, $P_k \in [P_{\min}, P_{\max}]$, is assigned in the kth busy period; the decisions are possibly random. Zero service rates are given in idle periods. The maximum cost $P_{\max}$ is assumed finite, but sufficiently large to ensure feasibility of required delay constraints. The minimum cost $P_{\min}$ is assumed large enough so that the queue is stable even if $P_{\min}$ is used for all time. Namely, we need $\sum_{n=1}^N \lambda_n \mathbb{E}[S_n] / \mu(P_{\min}) < 1$, or $\mu(P_{\min}) > \sum_{n=1}^N \lambda_n \mathbb{E}[S_n]$.

III. Convex Delay Optimization

We solve the convex delay minimization problem:

$$\text{minimize} \quad \sum_{n=1}^N f_n(\overline{W}_n) \tag{2}$$

$$\text{subject to} \quad \overline{W}_n \le d_n, \quad n = 1, \ldots, N \tag{3}$$

over scheduling policies described in Assumption 1. The functions f_n are continuous, convex, nondecreasing, and nonnegative for all n. Here, we assume a constant service rate, and that all delay constraints are feasible.

A. Delay Proportional Fairness

We say a delay vector $(\overline{W}_n^*)_{n=1}^N$ is *weighted delay proportional fair* if it is optimal under quadratic penalty functions $f_n(\overline{W}_n) = \frac{1}{2} c_n (\overline{W}_n)^2$ for all n, where c_n are given positive constants. In this case, any feasible delay vector $(\overline{W}_n)_{n=1}^N$ necessarily satisfies

$$\sum_{n=1}^N f_n'(\overline{W}_n^*)(\overline{W}_n - \overline{W}_n^*) = \sum_{n=1}^N c_n (\overline{W}_n - \overline{W}_n^*) \overline{W}_n^* \ge 0, \tag{4}$$

which is in the same spirit as *rate proportional fairness* [4]

$$\sum_{n=1}^N c_n \frac{x_n - x_n^*}{x_n^*} \le 0, \tag{5}$$

where $(x_n)_{n=1}^N$ is a feasible rate vector and $(x_n^*)_{n=1}^N$ is the optimal rate vector. Intuitively, delay proportional fairness has the product form (4) rather than the ratio form (5) because we favor large rates but want small delays. To further clarify, we give a two-user example showing that (4) and (5) provide the same proportional tradeoff. Let $c_1 = c_2 = 1$. In rate proportional fairness, consider two feasible rates $x_1 = 100$

and $x_2 = 10$; user 2 is 10 times worse than user 1. Then, if user 1 wants to increase x units of rate, it cannot cause user 2 more than $x/10$ units of loss. In delay proportional fairness, we suppose $\overline{W}_1 = 10$ and $\overline{W}_2 = 100$; user 2 is again 10 times worse than user 1. Then, according to (4), if user 1 wants to decrease delay by x units, user 2 can only tolerate up to $x/10$ units of delay increase, since user 2 is 10 times worse.

B. Delay Fairness Policy

For each user $n \in \{1, \ldots, N\}$, we define two discrete-time *virtual delay queues* $\{Z_{n,k}\}_{k=0}^{\infty}$ and $\{Y_{n,k}\}_{k=0}^{\infty}$, where $Z_{n,k+1}$ and $Y_{n,k+1}$ are computed at time t_{k+1} according to

$$Z_{n,k+1} = \max\left[Z_{n,k} + \sum_{i \in A_{n,k}} \left(W_{n,k}^{(i)} - d_n\right), 0\right], \quad (6)$$

$$Y_{n,k+1} = \max\left[Y_{n,k} + \sum_{i \in A_{n,k}} \left(W_{n,k}^{(i)} - r_{n,k}\right), 0\right], \quad (7)$$

where $W_{n,k}^{(i)}$ denote queueing delays of class n jobs served in the previous frame $[t_k, t_{k+1})$, and $r_{n,k} \in [0, d_n]$ are auxiliary variables chosen at time t_k independent of frame size T_k and per-frame class n arrivals $A_{n,k}$. For each virtual arrival $W_{n,k}^{(i)}$ at both queues, we match a virtual service d_n and $r_{n,k}$ in the $Z_{n,k}$ and $Y_{n,k}$ queue, respectively. Assume initially $Z_{n,0} = Y_{n,0} = 0$ for all n. Stabilizing the $Y_{n,k}$ queues helps to minimize convex delay penalty (2) (details omitted due to space limits). The values of $Z_{n,k}$ reflect the amount of past delays in class n exceeding the required delay bound d_n. We visualize that if $Z_{n,k}$ is kept small in the long run, then the average delay constraint $\overline{W}_n \leq d_n$ should be met. This is formalized by the next lemma, which shows that the stability of the $Z_{n,k}$ queue achieves the delay constraint $\overline{W}_n \leq d_n$.

Definition 1. We say queue $Z_{n,k}$ is mean rate stable if $\lim_{K\to\infty} \mathbb{E}\left[Z_{n,K}\right]/K = 0$.

Lemma 1. If queue $Z_{n,k}$ is mean rate stable, then $\overline{W}_n \leq d_n$.

The next policy solves the convex optimization (2)-(3).

Delay Fairness Policy (DelayFair)

1) In every frame k, prioritize job classes in the decreasing order of the ratio $(Z_{n,k} + Y_{n,k})/\mathbb{E}\left[S_n\right]$, where $\mathbb{E}\left[S_n\right]$ is the mean size of a class n job; ties are broken arbitrarily.
2) At the end of frame k, compute $Z_{n,k+1}$ and $Y_{n,k+1}$ for all classes n by (6) and (7), respectively, where $r_{n,k}$ is the solution to the convex program:

$$\text{minimize} \quad V f_n(r_{n,k}) - Y_{n,k}\,\lambda_n\, r_{n,k} \quad (8)$$

$$\text{subject to} \quad 0 \leq r_{n,k} \leq d_n \quad (9)$$

where $V > 0$ is a predefined control parameter.

The DelayFair policy requires arrival rates λ_n and mean job sizes $\mathbb{E}\left[S_n\right]$, but not higher-order statistics. The problem (8)-(9) is easily solved when $f_n(\cdot)$ is differentiable.

For pure feasibility problems that only seek to achieve all delay constraints $\overline{W}_n \leq d_n$ (so that $f_n = 0$ for all n in (2)-(3)), we have a different algorithm called DelayFeas. This does not use the $Y_{n,k}$ queues and does not require any statistics for arrivals and job sizes.

Delay Feasible Policy (DelayFeas)

- Prioritize job classes in the decreasing order of $Z_{n,k}$ in every busy period.
- Update $Z_{n,k}$ for all n at the end of every busy period.

C. Performance of DelayFair and DelayFeas

Theorem 1. Given any delay bounds $\{d_1, \ldots, d_N\}$ that are feasible under the class of scheduling policies defined in Assumption 1, both DelayFair and DelayFeas policy stabilize all $Z_{n,k}$ queues in the mean rate stable sense, and therefore satisfy delay constraints $\overline{W}_n \leq d_n$ for all n (by Lemma 1). In addition, the DelayFair policy yields convex delay cost satisfying

$$\limsup_{K\to\infty} \sum_{n=1}^{N} f_n\left(\frac{\mathbb{E}\left[\sum_{k=0}^{K-1}\sum_{i\in A_{n,k}} W_{n,k}^{(i)}\right]}{\mathbb{E}\left[\sum_{k=0}^{K-1}|A_{n,k}|\right]}\right) \leq \frac{C\sum_{n=1}^{N}\lambda_n}{V} + \sum_{n=1}^{N} f_n(\overline{W}_n^*),$$

where $V > 0$ is a predefined control parameter, $C > 0$ a finite constant, and the vector $(\overline{W}_n^*)_{n=1}^N$ is the optimal solution to the problem (2)-(3). The convex delay cost can be made arbitrarily close to the optimal $\sum_{n=1}^N f_n(\overline{W}_n^*)$ by a sufficiently large V.

IV. Delay-Constrained Optimal Rate Control

In this section, we incorporate dynamic allocations of the service rate. As specified in Assumption 2, we focus on frame-based policies that allocate a fixed service rate $\mu(P_k)$ in the kth busy period. Here, the frame size T_k, busy period B_k, per-frame class n arrivals $A_{n,k}$, and queueing delays $W_{n,k}^{(i)}$, all depend on $\mu(P_k)$ or cost P_k. Similar to (1), we define the average service cost

$$\overline{P} \triangleq \limsup_{K\to\infty} \frac{\mathbb{E}\left[\sum_{k=0}^{K-1} P_k\, B_k(P_k)\right]}{\mathbb{E}\left[\sum_{k=0}^{K-1} T_k(P_k)\right]}, \quad (10)$$

where $B_k(P_k)$ and $T_k(P_k)$ emphasize the dependence of B_k and T_k on P_k. It is easy to show that $B_k(P_k)$ and $T_k(P_k)$ are decreasing in P_k. The goal is to solve the delay-constrained optimal rate control problem:

$$\text{minimize:} \quad \overline{P} \quad (11)$$

$$\text{subject to:} \quad \overline{W}_n \leq d_n, \quad n = 1, \ldots, N \quad (12)$$

over control policies defined in Assumption 1 and 2.

A. Dynamic Rate Control Policy

We set up the same virtual queues $Z_{n,k}$ as in (6). Assume initially $Z_{n,0} = 0$ for all n.

*Dynamic Rate Control (*DynRate*) Policy*

1) In frame $k \in \mathbb{Z}^+$, use the strict priority policy π_k that prioritizes job classes in the decreasing order of $Z_{n,k}/\mathbb{E}[S_n]$, where $\mathbb{E}[S_n]$ is the mean job size of class n; ties are broken arbitrarily.
2) In the busy period of frame $k \in \mathbb{Z}^+$, allocate service rate $\mu(P_k)$, where P_k is the solution to

$$\text{minimize} \quad \left(V \sum_{n=1}^{N} \lambda_n \mathbb{E}[S_n]\right) \frac{P_k}{\mu(P_k)} + \sum_{n=1}^{N} Z_{n,k}\, \lambda_n\, \overline{W}_n(\pi_k, P_k) \tag{13}$$

$$\text{subject to} \quad P_k \in [P_{\min}, P_{\max}], \tag{14}$$

where $V > 0$ is a predefined control parameter. The term $\overline{W}_n(\pi_k, P_k)$ is the average queueing delay of class n when service rate $\mu(P_k)$ and strict priority policy π_k are used in all busy periods. If job classes are properly re-ordered according to policy π_k so that

$$\frac{Z_{1,k}}{\mathbb{E}[S_1]} \geq \cdots \geq \frac{Z_{N,k}}{\mathbb{E}[S_N]},$$

then the average delay $\overline{W}_n(\pi_k, P_k)$ is given by [19]

$$\overline{W}_n(\pi_k, P_k) = \frac{\frac{1}{2}\sum_{i=1}^{N} \lambda_i \mathbb{E}\left[X_i^2\right]}{(1 - \sum_{i=0}^{n-1} \rho_i)(1 - \sum_{i=0}^{n} \rho_i)},$$

where the random variable $X_i \triangleq S_i/\mu(P_k)$ denotes the service time of a class i job in frame k, $\rho_i \triangleq \lambda_i \mathbb{E}[X_i]$ for $i = 1, \ldots, N$, and $\rho_0 \triangleq 0$.
3) Update all $Z_{n,k}$ queues by (6) at every frame boundary.

The DynRate policy needs the knowledge of arrival rates and the first two moments of job sizes. To remove its dependence on second-order statistics, we can divide (13) by the (unknown) constant $\widetilde{R} \triangleq \frac{1}{2}\sum_{n=1}^{N} \lambda_n \mathbb{E}\left[S_n^2\right]$ and redefine $\widetilde{V} = V/\widetilde{R}$. The modified policy has the same performance (given in Theorem 2 below) as the DynRate policy, but only depends on first-order statistics.

B. Performance of DynRate

Theorem 2. The DynRate policy satisfies delay constraints $\overline{W}_n \leq d_n$ for all classes n and yields average cost $\overline{P}$ satisfying

$$\overline{P} \leq \frac{C \sum_{n=1}^{N} \lambda_n}{V} + P^*,$$

where P^* is the optimal average cost in (11)-(12), $C > 0$ is a finite constant, and $V > 0$ is a predefined control parameter. The gap between $\overline{P}$ and P^* can be made arbitrarily small by a sufficiently large V.

V. Simulations

We simulate the DelayFair, DelayFeas, and DynRate policy in a two-class nonpreemptive $M/G/1$ queue. Let $\mathcal{W}(P)$ be the queueing delay region under a constant service rate $\mu(P)$. Define $\rho_n \triangleq \lambda_n \mathbb{E}[X_n]$ and $R \triangleq \frac{1}{2}\sum_{n=1}^{2} \lambda_n \mathbb{E}\left[X_n^2\right]$, where $X_n = S_n/\mu(P)$ is service time of a class n job. We have [7]

$$\mathcal{W}(P) = \left\{ (\overline{W}_1, \overline{W}_2) \;\middle|\; \begin{array}{l} \overline{W}_1 \geq \dfrac{R}{1-\rho_1},\ \overline{W}_2 \geq \dfrac{R}{1-\rho_2}, \\ \rho_1 \overline{W}_1 + \rho_2 \overline{W}_2 = \dfrac{(\rho_1+\rho_2)R}{1-\rho_1-\rho_2} \end{array} \right\}. \tag{15}$$

In (15), the two inequalities show that the mean queueing delay in one class is minimized when it has priority over the other class. The equality is the $M/G/1$ conservation law [20].

Every simulation result in this section is a sample average over 10 runs, each of which lasts for 10^6 frames.

A. Simulation for DelayFair and DelayFeas

We consider a two-class $M/M/1$ queue with arrival rates $(\lambda_1, \lambda_2) = (1, 2)$ and mean service times $(\mathbb{E}[X_1], \mathbb{E}[X_2]) = (0.4, 0.2)$. Here we assume a constant service rate. The performance region $\mathcal{W}$ of mean queueing delay, using (15), is

$$\mathcal{W} = \left\{ (\overline{W}_1, \overline{W}_2) \;\middle|\; \overline{W}_1 + \overline{W}_2 = 2.4, \overline{W}_1 \geq 0.4, \overline{W}_2 \geq 0.4 \right\}.$$

1) The DelayFair *Policy:* We consider the delay proportional fairness problem:

$$\text{minimize:} \quad f(\overline{W}_1, \overline{W}_2) = 0.5\,(\overline{W}_1)^2 + 2\,(\overline{W}_2)^2 \tag{16}$$

$$\text{subject to:} \quad (\overline{W}_1, \overline{W}_2) \in \mathcal{W},\ \overline{W}_1 \leq 1.95,\ \overline{W}_2 \leq 1. \tag{17}$$

The optimal solution to (16)-(17) is $(\overline{W}_1^*, \overline{W}_2^*) = (1.92, 0.48)$. We simulate the DelayFair policy for different values of control parameter V, with results in Table I. The values in parentheses are sample standard deviations over the 10 simulation runs. As V increases, DelayFair yields near-optimal delay penalty.

V	$\overline{W}_1$	$\overline{W}_2$	$f(\overline{W}_1, \overline{W}_2)$
100	1.6607 (0.0055)	0.7424 (0.0052)	2.4814 (0.0239)
1000	1.7977 (0.0057)	0.5984 (0.0043)	2.3321 (0.0199)
2000	1.8339 (0.0056)	0.5639 (0.0053)	2.3176 (0.0217)
5000	1.8679 (0.0073)	0.5276 (0.0050)	2.3014 (0.0222)
Optimal:	1.92	0.48	2.304

TABLE I
Simulation for the DelayFair policy for delay proportional fairness under different values of V.

2) The DelayFeas *Policy:* We perform five sets of simulations to achieve delay constraints $\overline{W}_n \leq d_n$ for $n \in \{1, 2\}$ using $(d_1, d_2) = (0.45, 2.05)$, $(0.85, 1.65)$, $(1.25, 1.25)$, $(1.65, 0.85)$, and $(2.05, 0.45)$; they are all $(0.05, 0.05)$ entry-wise larger than a feasible point on $\mathcal{W}$. Fig. 1 shows that the DelayFeas policy adaptively yields feasible average delays in response to different constraints. Over the 10 simulation runs in each of the five cases, the sample standard deviation of the average delay in each job class is at most 0.017; all simulation runs produce consistent results.

B. Simulation for the DynRate policy

We consider a two-class $M/G/1$ queue with arrival rates $(\lambda_1, \lambda_2) = (1, 2)$. The size of a class 1 job is 0.5 with probability 0.8, and 3 otherwise. The size of a class 2 job is

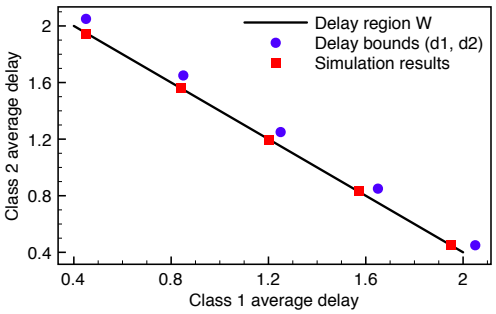

Fig. 1. Simulation for the DelayFeas policy to achieve delay constraints $\overline{W}_n \leq d_n$ for $n \in \{1, 2\}$ under different (d_1, d_2).

always one. We consider the service rate $\mu(P) = \sqrt{P}$, where P takes values in the discrete set $\{16, 25\}$.

Under dynamic rate control, the full queueing delay region, denoted by $\mathcal{W}$, is the convex hull of the two individual delay regions $\mathcal{W}(16)$ and $\mathcal{W}(25)$ defined in (15) (see Fig. 2), where $\mathcal{W}(16)$ and $\mathcal{W}(25)$ are the queueing delay region under a constant service cost of $P = 16$ and 25, respectively.

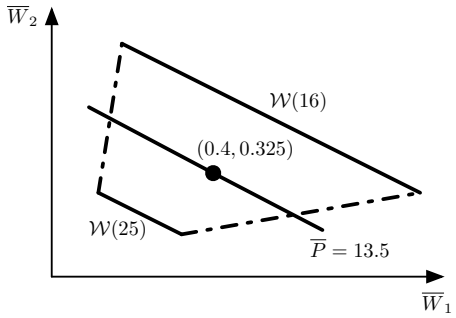

Fig. 2. The performance region $\mathcal{W}$ of mean queueing delay in the simulation for the DynRate policy.

We use the DynRate policy to solve:

$$\text{minimize:} \quad \overline{P} \tag{18}$$

$$\text{subject to:} \quad (\overline{W}_1, \overline{W}_2) \in \mathcal{W} \tag{19}$$

$$\overline{W}_1 \leq 0.4, \ \overline{W}_2 \leq 0.325 \tag{20}$$

where $\mathcal{W}$ is the full delay region in Fig. 2. The minimum cost is achieved by satisfying (20) with equality. By finding the optimal stationary randomized policy that yields $(\overline{W}_1, \overline{W}_2) = (0.4, 0.325)$, we know the minimum average cost is 13.5. Table II shows that the average cost and average delay in each job class under the DynRate policy approaches optimality with the increase of control parameter V.

V	$\overline{W}_1$	$\overline{W}_2$	$\overline{P}$
1	0.3562 (0.00078)	0.3029 (0.00032)	13.802 (0.018)
10	0.3984 (0.00022)	0.3247 (0.00005)	13.510 (0.026)
100	0.4003 (0.00013)	0.3252 (0.00010)	13.504 (0.022)
Optimal:	0.4	0.325	13.5

TABLE II
SIMULATION FOR THE DynRate POLICY UNDER DIFFERENT VALUES OF V

VI. DISCUSSIONS

The adaptive method introduced in this paper implies that convex optimization problems in multi-class queues can be reduced to solving a sequence of linear cost minimization problems, one for each renewal period. In an $M/G/1$ queue, these linear cost problems are solved by the $c\mu$ rule because the delay region is a polymatroid. There are many other queueing systems having polymatroid-type performance regions (see [2], [3] for examples), in which linear costs are minimized by strict priority strategies. Our method may be applicable to solving interesting convex optimization problems and developing online dynamic priority policies there.

REFERENCES

[1] D. Bertsimas, "The achievable region method in the optimal control of queueing systems; formulations, bounds, and policies," *Queueing Syst.*, vol. 21, no. 3-4, pp. 337–389, Sep. 1995.

[2] D. Bertsimas and J. Niño-Mora, "Conservation laws, extended polymatroids, and multiarmed bandit problems; a polyhedral approach to indexable systems," *Math. of Oper. Res.*, vol. 21, no. 2, pp. 257–306, May 1996.

[3] D. D. Yao, "Dynamic scheduling via polymatroid optimization," in *Performance Evaluation of Complex Systems: Techniques and Tools, Performance 2002, Tutorial Lectures*. London, UK: Springer-Verlag, 2002, pp. 89–113.

[4] F. P. Kelly, "Charging and rate control for elastic traffic," *European Trans. Telecommunications*, vol. 8, pp. 33–37, 1997.

[5] W.-H. Wang, M. Palaniswami, and S. H. Low, "Application-oriented flow control: Fundamentals, algorithms, and fairness," *IEEE/ACM Trans. Netw.*, vol. 14, no. 6, pp. 1282–1291, Dec. 2006.

[6] N. Bansal, T. Kimbrel, and K. Pruhs, "Speed scaling to manage energy and temperature," *Journal of the ACM*, vol. 54, no. 1, Mar. 2007.

[7] E. Gelenbe and I. Mitrani, *Analysis and Synthesis of Computer Systems*, 2nd ed. Imperial College Press, 2010.

[8] C.-P. Li and M. J. Neely, "Network utility maximization over partially observable Markovian channels," in *IEEE Proc. Int. Symp. Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, Princeton, NJ, USA, May 2011.

[9] M. J. Neely, *Stochastic Network Optimization with Application to Communication and Queueing Systems*. Morgan & Claypool, 2010.

[10] C.-P. Li, "Stochastic optimization over parallel queues: Channel-blind scheduling, restless bandit, and optimal delay," Ph.D. dissertation, University of Southern California, 2011.

[11] E. Altman and A. Shwartz, "Optimal priority assignment: a time sharing approach," *IEEE Trans. Autom. Control*, vol. 34, no. 10, pp. 1098 –1102, Oct. 1989.

[12] K. W. Ross and B. Chen, "Optimal scheduling of interactive and non-interactive traffic in telecommunication systems," *IEEE Trans. Autom. Control*, vol. 33, no. 3, pp. 261–267, Mar. 1988.

[13] A. Federgruen and H. Groenevelt, "Characterization and optimization of achievable performance in general queueing systems," *Oper. Res.*, vol. 36, no. 5, pp. 733–741, 1988.

[14] P. P. Bhattacharya, L. Georgiadis, and P. Tsoucas, "Problems of adaptive optimization in multiclass M/GI/1 queues with bernoulli feedback," *Math. of Oper. Res.*, vol. 20, no. 2, pp. 355–380, May 1995.

[15] P. P. Bhattacharya, L. Georgiadis, P. Tsoucas, and I. Viniotis, "Adaptive lexicographic optimization in multi-class M/GI/1 queues," *Math. of Oper. Res.*, vol. 18, no. 3, pp. 705–740, Aug. 1993.

[16] J. M. George and J. M. Harrison, "Dynamic control of a queue with adjustable service rate," *Oper. Res.*, vol. 49, no. 5, pp. 720–731, 2001.

[17] S. Stidham and R. Weber, "Monotonic and insensitive optimal policies for contorl of queues with undiscounted costs," *Oper. Res.*, vol. 37, no. 4, pp. 611–625, 1989.

[18] M. J. Neely, "Dynamic optimization and learning for renewal systems," in *Asilomar Conf. Signals, Systems, and Computers*, Nov. 2010.

[19] D. P. Bertsekas and R. G. Gallager, *Data Networks*, 2nd ed. Prentice Hall, 1992.

[20] L. Kleinrock, *Queueing Systems*. Wiley Interscience, 1976, vol. II: Computer Applications.