Dynamic Optimization and Learning for Renewal Systems

Michael J. Neely

***Abstract*—This paper considers optimization of time averages in systems with variable length renewal frames. Applications include power-aware and profit-aware scheduling in wireless networks, peer-to-peer networks, and transportation systems. Every frame, a new policy is implemented that affects the frame size and that creates a vector of attributes. The policy can be a single decision in response to a random event observed on the frame, or a sequence of such decisions. The goal is to choose policies on each frame in order to maximize a time average of one attribute, subject to additional time average constraints on the others. Two algorithms are developed, both based on Lyapunov optimization concepts. The first makes decisions to minimize a "drift-plus-penalty" ratio over each frame. The second is similar but does not involve a ratio. For systems that make only a single decision on each frame, both algorithms are shown to learn efficient behavior without a-priori statistical knowledge. The framework is also applicable to large classes of constrained Markov decision problems. Such problems are reduced to finding an approximate solution to a simpler unconstrained stochastic shortest path problem on each frame. Approximations for the simpler problem may still suffer from a curse of dimensionality for systems with large state space. However, our results are compatible with any approximation method, and demonstrate an explicit tradeoff between performance and convergence time.**

***Index Terms*—Stochastic processes, Markov decision problems**

I. Introduction

Consider a stochastic system that regularly experiences times when the system state is refreshed, called *renewal times*. The goal is to develop a control algorithm that maximizes the time average of a reward process associated with the system, subject to time average constraints on a collection of penalty processes. The renewal-reward theorem is a simple and elegant technique for computing time averages in such systems (see, for example, [2][3]). Using basic renewal-reward theory, it can be shown that there exists an optimal algorithm that makes independent and identically distributed (i.i.d.) decisions over each renewal frame. However, it is typically difficult to choose in advance such a policy, because this would require a-priori knowledge of all system probabilities, and would require some computation method for finding the i.i.d. policy based on this knowledge. This paper designs dynamic algorithms that use different control policies on each frame, based on learning from the past, and shows the performance of these algorithms is very close to that of the best i.i.d. algorithm. Specifically, the dynamic algorithms can be parameterized by a constant ϵ, and are shown to achieve a time average reward within $O(\epsilon)$ of optimal, for any desired $\epsilon > 0$, with a tradeoff in convergence time that is polynomial in $1/\epsilon$.

This renewal problem arises in many different applications. One application of interest is a *task processing network*. For example, consider a network of wireless devices that repeatedly collaborate to accomplish tasks (such as reporting sensor data to a destination, performing distributed computation on data, or coordinating peer-to-peer downloads). Tasks are performed one after the other, and for each task we must decide what modes of operation and communication to use, possibly allowing some nodes of the network to remain idle to save power. It is then important to make decisions that maximize the time average utility associated with task processing, subject to time average power constraints at each node. The renewal framework also applies to transportation systems. We consider a simple example where a taxi driver repeatedly decides to take either 1 or 2 customers, based on their requested destinations, in order to maximize time average profit subject to average delay constraints.

This paper develops a general framework for solving such problems. To do so, we extend the theory of Lyapunov optimization from [4][5]. Specifically, work in [4][5] considers discrete time queueing networks and develops a simple *drift-plus-penalty* rule for making optimal decisions. These decisions are made in a greedy manner every slot based only on the observed traffic and channel conditions for that slot, without requiring a-priori knowledge of the underlying probability distribution. However, the work in [4][5] assumes that all slots have fixed length, that there is a single random event that is observed at the beginning of each slot, and that only a single decision is made on that slot based on this observation.

The renewal problem treated in this paper is more complex because each frame may have a different length and may contain a sequence of random events. This requires selection of a *decision policy* for each frame, often involving a sequence of decisions rather than a single decision. We develop two algorithms for such systems, both based on Lyapunov optimization. The first chooses a policy on each frame to minimize a weighted *drift-plus-penalty ratio*. The second is similar but does not involve a ratio. Both algorithms can be implemented without a-priori statistical knowledge for systems that involve *single decision policies*. For systems that involve *sequential decision policies*, such as Markov Decision Problems (MDPs), the algorithms reduce to solving an unconstrained weighted stochastic shortest path problem on each frame. The unconstrained problem may still be difficult

Michael J. Neely is with the Electrical Engineering department at the University of Southern California, Los Angeles, CA.

This paper was presented in part at the Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, CA, 2010 [1]. This material is supported in part by one or more of the following: the NSF Career grant CCF-0747525, the Network Science Collaborative Technology Alliance sponsored by the U.S. Army Research Laboratory W911NF-09-2-0053.

to solve for Markov systems with large dimension. However, we show that performance scales gracefully with *approximate* solutions. This allows our results to be used in conjunction with any approximation method for shortest paths, such as those based on approximate dynamic programming, neuro-dynamic programming, or q-learning [6][7].

In Section III we show that, in cases when policies involve single decisions, the optimal algorithm can in principle be written as a *linear fractional program* with a possibly uncountably infinite number of parameters and decision variables. Offline approaches exist for solving linear fractional programs with a finite number of parameters and variables. For example, [8] uses a non-linear change of variables to transform a (non-convex) linear fractional program into a convex program. However, this does not generalize to online approaches, where we must optimize averages without necessarily knowing the system probabilities. This is because time averages are not preserved in the non-linear transformation. Thus, our approach can also be viewed as a new *online* algorithm for linear fractional programs. We believe this is the first of its kind.

A. Relation to Prior Work

Our reduction to an unconstrained problem uses a *virtual queue* that is conceptually related to Lagrange multipliers for constrained MDPs. For example, theorems in [9][10] show that an optimal control policy for a constrained MDP satisfies a Bellman equation for a corresponding unconstrained MDP, where the unconstrained problem is parameterized by optimal Lagrange multipliers. However, we do not know what the optimal Lagrange multipliers are. Further, even if the optimal Lagrange multipliers and the optimal value function for the corresponding Bellman equation were known, this information cannot necessarily help to solve the constrained problem. This is because the Bellman equation may have an infinite number of solutions, necessarily including the optimal policy, but often including other policies that are not optimal.[1]

Despite this shortcoming, works on constrained MDPs in [11][12][13][14][15][16] use Lagrange multiplier *update rules* that dynamically approximate the optimal multipliers. The work in [11][12] provides an asymptotically optimal technique based on these updates using a fluid limit and a multi-timescale analysis. The technique works because the dynamic updates reach a fluid steady state that can be interpreted in terms of time average constraint satisfaction. Such dynamics are crucial, and the algorithm would not necessarily satisfy the constraints if the Lagrange approximations were replaced by the exact multipliers that are being approximated. The Lagrange update rules in [11][12][13][14][15][16] are similar to our virtual queue updates. However, our use of the virtual queue methodology provides more direct physical intuition, and yields simplified proofs that do not require the multi-timescale and fluid model transformations used in [11][12], or the stochastic approximation lemmas and sub-results used in [13][14][15][16]. Because of this, we obtain a direct bound on the average virtual queue size at any time, which specifies the degree to which the constraints are violated over any interval of time. This provides insight into the convergence time required to yield an ϵ-optimal solution.

[1]Similarly, if one knows the optimal Lagrange multipliers for a linear program, one can solve an unconstrained dual problem to obtain the optimal objective value, but the resulting primal variables may not satisfy the desired constraints.

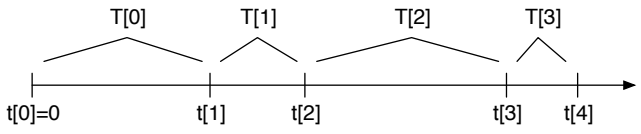


Fig. 1. A timeline illustrating renewal frames for the system.

An alternative approach to solving constrained MDPs is to map the problem to a linear program with one variable for each state-action pair (see, for example, [3][9]). However, this is an offline computation that requires the state and action space to be finite, and also requires a-priori knowledge of all system probabilities.

The drift-plus-penalty ratio technique in this paper leverages ideas in [17][18], where [17] considers a frame-based Lyapunov framework for Markov decision problems involving network delay, and [18] develops a ratio rule for utility optimization in restless bandit systems. Recent work in [19] uses a renewal model for a task processing system, where multiple wireless "reporting nodes" select data formats (e.g., "voice" or "video") in which to deliver sensed information. However, the renewal times in [19] are not influenced by control decisions. More general problems can be treated using the theory developed in the current paper.

This paper assumes tasks are processed one at a time, and does not treat systems where tasks are pipelined through multiple stages, where a new task can be started while an old task is still being processed elsewhere. Multi-stage task processing is treated in [20][21] for the case when each stage involves an action that takes a fixed length time slot (rather than a variable frame length).

B. Outline of Paper

The next section specifies the renewal model. Section III shows the relation to linear fractional problems. Section IV develops the drift-plus-penalty ratio algorithm. Section V shows how to minimize the drift-plus-penalty ratio using a bi-section procedure. Section VI provides an alternative algorithm that does not involve a ratio. Section VII provides specific examples for wireless networks and transportation systems.

II. Renewal System Model

Suppose the set of non-negative real numbers is segmented into successive frames of duration $\{T[0], T[1], T[2], \ldots\}$, as shown in Fig. 1. Define $t[0] = 0$, and for each positive integer r define $t[r]$ as the *rth renewal time*:

$$t[r] \triangleq \sum_{i=0}^{r-1} T[i]$$

The interval of all times t such that $t[r] \leq t < t[r+1]$ is defined as the *rth renewal frame*, defined for each $r \in \{0, 1, 2, \ldots\}$.

Let $\mathcal{P}$ be an abstract (possibly uncountably infinite) set. At the beginning of each renewal frame r, the controller selects an element $\pi[r] \in \mathcal{P}$, and this choice results in a $(L+2)$-dimensional random vector $(T[r], y_0[r], y_1[r], \ldots, y_L[r])$ (where L is some non-negative integer). The value $T[r]$ is the (positive) size of frame r, and the value $y_l[r]$ is the value of the lth *penalty* for frame r. Penalties can be positive or negative, representing, for example, power expenditures incurred in some components of a network, or negative rewards earned. The system is completely specified by the set $\mathcal{P}$ and a random function that maps each element $\pi \in \mathcal{P}$ to a random vector $(\hat{T}(\pi), \hat{y}_0(\pi), \ldots, \hat{y}_L(\pi))$, where the vector has a distribution that depends on π. Thus, the vector for frame r is defined by $\pi[r]$ as follows:

$$(T[r], y_0[r], y_1[r], \ldots, y_L[r]) \triangleq (\hat{T}(\pi[r]), \hat{y}_0(\pi[r]), \ldots, \hat{y}_L(\pi[r])) \quad (1)$$

The vector for frame r is assumed to be conditionally independent of all past events, given the current $\pi[r]$ used. The entries are assumed to have conditional first and second moments that are bounded regardless of the $\pi[r]$ used. That is, there are finite constants T^{min}, T^{max}, y_0^{min}, y_0^{max}, d such that for all $\pi[r] \in \mathcal{P}$ and all $l \in \{0, 1, \ldots, L\}$ we have:

$$y_0^{min} \leq \mathbb{E}\left[\hat{y}_0(\pi[r])|\pi[r]\right] \leq y_0^{max} \quad (2)$$

$$0 < T^{min} \leq \mathbb{E}\left[\hat{T}(\pi[r])|\pi[r]\right] \leq T^{max} \quad (3)$$

$$\mathbb{E}\left[\hat{T}(\pi[r])^2|\pi[r]\right] \leq d \ , \ \mathbb{E}\left[\hat{y}_l(\pi[r])^2|\pi[r]\right] \leq d \quad (4)$$

The above model formally treats π as an element in a set $\mathcal{P}$. We shall call such an element π a *policy*, and call the set $\mathcal{P}$ a *policy space*. This terminology fits the intuitive description of the system: Imagine a stochastic system whose behavior is affected by the manner in which it is operated. Every new frame r, the controller specifies how to operate the system by choosing a *policy* $\pi[r] \in \mathcal{P}$. The random mapping from $\pi[r]$ to an output vector $(\hat{T}(\pi[r]), \hat{y}_0(\pi[r]), \ldots, \hat{y}_L(\pi[r]))$ corresponds to an independent simulation of the sample path dynamics of the system under policy $\pi[r]$. The output vector is determined from this sample path. Such a description of course only makes sense if the system dynamics result in an output vector that is probabilistically well defined and measurable. We give three example policy structures below.

A. Single Decision Policies with Observed Initial Information

Single decision policies have the following structure: At the beginning of every frame r, the controller observes a random vector $\boldsymbol{\eta}[r]$ that specifies information about the current task. The sequence $\{\boldsymbol{\eta}[r]\}_{r=0}^{\infty}$ is assumed to be i.i.d. over frames. For each possible outcome $\boldsymbol{\eta}$, define $\mathcal{A}_{\boldsymbol{\eta}}$ as the *action space* associated with $\boldsymbol{\eta}$. Every frame, the controller observes $\boldsymbol{\eta}[r]$ and chooses a single action $\alpha[r] \in \mathcal{A}_{\boldsymbol{\eta}[r]}$. The pair $(\boldsymbol{\eta}[r], \alpha[r])$ determines the output vector. A *policy* π is a contingency plan for choosing a single action $\alpha \in \mathcal{A}_{\boldsymbol{\eta}}$ given that $\boldsymbol{\eta}$ is observed.[2]

[2]If $\boldsymbol{\eta}[r]$ can take values only in a finite set Ω, and if $\mathcal{A}_{\boldsymbol{\eta}}$ is finite for all $\boldsymbol{\eta} \in \Omega$, then a policy π is described by a collection of conditional probabilities $Pr[\alpha[r] = \alpha | \boldsymbol{\eta}[r] = \boldsymbol{\eta}]$ for all $\boldsymbol{\eta} \in \Omega$ and all $\alpha \in \mathcal{A}_{\boldsymbol{\eta}}$.

The output vector is a random function of π. However, in this case it can be written explicitly in terms of $\boldsymbol{\eta}[r]$ and $\alpha[r]$:

$$(T[r], y_0[r], \ldots, y_L[r]) \triangleq (\tilde{T}(\boldsymbol{\eta}[r], \alpha[r]), \tilde{y}_0(\boldsymbol{\eta}[r], \alpha[r]), \ldots, \tilde{y}_L(\boldsymbol{\eta}[r], \alpha[r]))$$

where $\tilde{T}(\boldsymbol{\eta}[r], \alpha[r])$ and $\tilde{y}_l(\boldsymbol{\eta}[r], \alpha[r])$ are *deterministic functions* of $\boldsymbol{\eta}[r]$ and $\alpha[r]$. A key challenge here is to develop an optimal algorithm without knowledge of the probability distribution for $\boldsymbol{\eta}[r]$. If the vector $\boldsymbol{\eta}[r]$ has K dimensions, and each component has 10^9 possible values, then there are 10^{9K} possibilities. It would be impossible to estimate the corresponding probabilities. Remarkably, the algorithms we develop do not require an estimation of this enormous number of parameters, yet they do converge quickly with low complexity.

This model has a close relationship to *opportunistic scheduling* in time-slotted wireless networks, where a random channel state vector is observed every slot before making a transmission decision. Prior work in [22] develops a Lyapunov drift rule for queue stability in these networks, and work in [5] extends this to a drift-plus-penalty rule for joint stability and penalty minimization. However, these cannot be used for systems with variable frame lengths. The *drift-plus-penalty ratio rule* in the current paper can be viewed as a generalization of both of these techniques, reducing to the drift-plus-penalty rule from [5] in the special case of time-slotted systems with penalty minimization, and to the drift rule from [22] for time-slotted systems without penalty minimization.

This single decision policy structure is simple but rich, and Section VII provides three specific examples. Without loss of continuity the reader can skip to that section for details on the examples and their corresponding algorithms.

B. Protocol Selection

In this scenario, a policy π corresponds to one of M pre-established *protocols* that can be used in a stochastic system, where M is a positive integer. For example, a policy space for a choice of three different routing protocols in a data network would be:

$$\mathcal{P} = \{\text{protocol 1, protocol 2, protocol 3}\}$$

This is an important example because networks are often pre-programmed with several protocol options, and our results provide an optimal means of switching between them. Specifically, every frame the controller selects a protocol $\pi \in \mathcal{P}$, which produces a random vector $(\hat{T}(\pi), \hat{y}_0(\pi), \ldots, \hat{y}_L(\pi))$ with distribution that depends on π, and that is conditionally independent of past vectors given the protocol π that is used. Unlike the structure in the previous section, where system optimality depends on the joint distribution of an observed vector $\boldsymbol{\eta}[r]$ of initial information, optimality in this scenario depends only on the marginals of the output vector. In fact, Lemma 1 in Section II-E can be used to show that, in this special case, optimality depends only on the averages of each vector component under each of the M protocols in $\mathcal{P}$:

$$\left(\mathbb{E}\left[\hat{T}(\pi)\right], \mathbb{E}\left[\hat{y}_0(\pi)\right], \ldots, \mathbb{E}\left[\hat{y}_L(\pi)\right]\right) \ \forall \pi \in \mathcal{P}$$

Such averages can either be assumed known, or can be accurately estimated using W samples of past output vectors for each $\pi \in \mathcal{P}$. Variance information is not required to implement the algorithms we develop. However, this information impacts the estimation error from W samples. It also impacts the convergence time of our algorithm through a constant B, defined later, that contains second moment information.

C. Sequential Decision Policies

Sequential decision policies have the following structure: Every frame r, the controller first observes a vector of information, then makes a decision, then observes another vector of information, and so on, for some finite number of decisions until the frame ends (where the end time might depend on the policy itself). The simplest specific example is when the entire system is defined by a discrete time MDP with a finite state space $\mathcal{S}$ that includes a "renewal" state 0, and a finite action space $\mathcal{A}$. The system is assumed to start in state 0 at the beginning of every frame, and all frame sizes are some positive integer number of slots. The transition probability matrix is $(P_{ij}(\alpha))$ for $i, j \in \mathcal{S}$ and $\alpha \in \mathcal{A}$. The renewal frame ends when the system returns to state 0. A policy π is a contingency plan for choosing actions on each slot of the frame based on the current state. The policy space $\mathcal{P}$ is the set of all such policies. To meet the boundedness assumptions, the MDP must have the property that state 0 is recurrent under any policy used, with bounded second moments of inter-renewal time. This can be enforced by the *forced renewal* concept in [17], which, every slot, randomly forces the system to return to state 0 with some probability p. A similar construction can be used for continuous time MDPs.

D. The Optimization Problem

We say that an *algorithm* is a method for choosing policies $\pi[r] \in \mathcal{P}$ over frames $r \in \{0, 1, 2, \ldots\}$. This paper restricts to causal algorithms that make decisions based only on past observations, without knowledge of the future. Consider any particular algorithm that chooses policies $\pi[r] \in \mathcal{P}$ every frame r according to some well defined (possibly probabilistic) rule. Define the following frame-average expectations for all positive integers R:

$$\overline{T}[R] \triangleq \frac{1}{R}\sum_{r=0}^{R-1} \mathbb{E}\left[T[r]\right] \ , \ \overline{y}_l[R] \triangleq \frac{1}{R}\sum_{r=0}^{R-1} \mathbb{E}\left[y_l[r]\right] \tag{5}$$

where $T[r]$ and $y_l[r]$ depend on the policy $\pi[r]$ by (1). Define the infinite horizon frame-average expectations $\overline{T}$, $\overline{y}_l$ by:

$$(\overline{T}, \overline{y}_l) = \lim_{R\to\infty} (\overline{T}[R], \overline{y}_l[R])$$

For simplicity, this subsection temporarily assumes the above limit is well defined ($\limsup$ is used in the main theorem). The goal is to choose a policy $\pi[r] \in \mathcal{P}$ every frame r to minimize the ratio $\overline{y}_0/\overline{T}$ subject to constraints on $\overline{y}_l/\overline{T}$:

$$\text{Minimize:} \quad \overline{y}_0/\overline{T} \tag{6}$$

$$\text{Subject to:} \quad \overline{y}_l/\overline{T} \leq c_l \ \forall l \in \{1, \ldots, L\} \tag{7}$$

$$\pi[r] \in \mathcal{P} \ \forall r \in \{0, 1, 2, \ldots\} \tag{8}$$

where c_l for $l \in \{1, \ldots, L\}$ are a given collection of real-valued (possibly negative) constants.

The motivation for looking at the ratio $\overline{y}_l/\overline{T}$ is that it defines the *time average penalty associated with the* $y_l[r]$ *process*. To understand this, suppose the following limits converge to constants y_l^{av} and T^{av} with probability 1:

$$\lim_{R\to\infty} \frac{1}{R}\sum_{r=0}^{R-1} y_l[r] = y_l^{av} \ , \ \lim_{R\to\infty} \frac{1}{R}\sum_{r=0}^{R-1} T[r] = T^{av} \ (w.p.1)$$

The second moment bounds (4) imply we can take expectations of the above and pass the expectations through the limits to conclude $\overline{y}_l = y_l^{av}$ and $\overline{T} = T^{av}$. Then the time average penalty per unit time associated with $y_l[r]$ (sampled only at renewal times for simplicity) satisfies with probability 1:

$$\lim_{R\to\infty} \frac{\sum_{r=0}^{R-1} y_l[r]}{\sum_{r=0}^{R-1} T[r]} = \lim_{R\to\infty} \frac{\frac{1}{R}\sum_{r=0}^{R-1} y_l[r]}{\frac{1}{R}\sum_{r=0}^{R-1} T[r]} = \frac{\overline{y}_l}{\overline{T}}$$

Therefore, the time average is equal to the ratio of frame averages. This observation is often used in *renewal-reward theory* [2][3]. In the special case when the policy $\pi[r]$ is chosen i.i.d. from the set $\mathcal{P}$ every frame r, then $\{T[r]\}_{r=0}^{\infty}$ and $\{y_l[r]\}_{r=1}^{\infty}$ are i.i.d. and the law of large numbers implies $\overline{y}_l/\overline{T} = \mathbb{E}\left[y_l[0]\right]/\mathbb{E}\left[T[0]\right]$ with probability 1.

For some problems, one may be more interested in optimizing the per-frame average $\overline{y}_0$, rather than the time average $\overline{y}_0/\overline{T}$. Such problems are easier and are treated in Section VI-A. Our work in [23][1] treats extensions of the problem (6)-(8) that seek to optimize convex functions of time averages, although this paper omits this extension for brevity.

E. Optimality of i.i.d. Algorithms

Here the problem (6)-(8) is stated more precisely using $\limsup$s, which do not require existence of a limit:

$$\text{Minimize:} \quad \limsup_{R\to\infty} \frac{\overline{y}_0[R]}{\overline{T}[R]} \tag{9}$$

$$\text{Subject to:} \quad \limsup_{R\to\infty} \frac{\overline{y}_l[R]}{\overline{T}[R]} \leq c_l \ \forall l \in \{1, \ldots, L\} \tag{10}$$

$$\pi[r] \in \mathcal{P} \ \forall r \in \{0, 1, 2, \ldots\} \tag{11}$$

Assume the constraints (10)-(11) are feasible, and define $ratio^{opt}$ as the infimum ratio in (9) over all algorithms that satisfy these constraints. The value $ratio^{opt}$ is finite by the boundedness assumptions (2)-(3).

Define an *i.i.d. algorithm* as one that, at the beginning of each new frame $r \in \{0, 1, 2, \ldots\}$, chooses a policy $\pi[r]$ by independently and probabilistically selecting $\pi \in \mathcal{P}$ according to some distribution that is the same for all frames r. Let $\pi^*[r]$ represent such an i.i.d. algorithm. Then the random variables $\{\hat{T}(\pi^*[r])\}_{r=0}^{\infty}$ are independent and identically distributed (i.i.d.) over frames, as are $\{\hat{y}_l(\pi^*[r])\}_{r=0}^{\infty}$.

Lemma 1: (Optimality over i.i.d. algorithms) If the constraints (10)-(11) are feasible and if the boundedness assumptions (2)-(3) hold, then for any $\delta > 0$, there exists an i.i.d. algorithm $\pi^*[r]$ that satisfies:

$$\mathbb{E}\left[\hat{y}_0(\pi^*[r])\right] \leq \mathbb{E}\left[\hat{T}(\pi^*[r])\right](ratio^{opt} + \delta) \tag{12}$$

$$\mathbb{E}\left[\hat{y}_l(\pi^*[r])\right] \leq \mathbb{E}\left[\hat{T}(\pi^*[r])\right](c_l + \delta) \ \forall l \in \{1, \ldots, L\} \tag{13}$$

Proof: See [24]. ■

III. Relation to Linear Fractional Programming

Consider the special case of single decision policies with initial information vectors $\boldsymbol{\eta}[r]$ that take values in a finite set Ω, with probabilities $q(\boldsymbol{\eta})$ for all $\boldsymbol{\eta} \in \Omega$. Assume the action space $\mathcal{A}$ is finite and does not depend on $\boldsymbol{\eta}$. Then a general i.i.d. algorithm is given by a *conditional probability distribution* $p(\alpha|\boldsymbol{\eta})$, being the probability that the controller chooses action $\alpha \in \mathcal{A}$, given it observes $\boldsymbol{\eta}[r] = \boldsymbol{\eta}$ on frame r. In principle, one could find the optimal i.i.d. algorithm by solving the following *linear fractional program*:

$$\text{Min.:} \quad \frac{\sum_{\boldsymbol{\eta}\in\Omega}\sum_{\alpha\in\mathcal{A}} q(\boldsymbol{\eta})p(\alpha|\boldsymbol{\eta})\tilde{y}_0(\boldsymbol{\eta},\alpha)}{\sum_{\boldsymbol{\eta}\in\Omega}\sum_{\alpha\in\mathcal{A}} q(\boldsymbol{\eta})p(\alpha|\boldsymbol{\eta})\tilde{T}(\boldsymbol{\eta},\alpha)}$$

$$\text{Subj. to:} \quad \frac{\sum_{\boldsymbol{\eta}\in\Omega}\sum_{\alpha\in\mathcal{A}} q(\boldsymbol{\eta})p(\alpha|\boldsymbol{\eta})\tilde{y}_l(\boldsymbol{\eta},\alpha)}{\sum_{\boldsymbol{\eta}\in\Omega}\sum_{\alpha\in\mathcal{A}} q(\boldsymbol{\eta})p(\alpha|\boldsymbol{\eta})\tilde{T}(\boldsymbol{\eta},\alpha)} \leq c_l \ \forall l \in \{1,\ldots,L\}$$

$$p(\alpha|\boldsymbol{\eta}) \geq 0 \ \forall \alpha \in \mathcal{A}, \ \forall \boldsymbol{\eta} \in \Omega$$

$$\sum_{\alpha\in\mathcal{A}} p(\alpha|\boldsymbol{\eta}) = 1 \ \forall \boldsymbol{\eta} \in \Omega$$

where the above program uses constants $q(\boldsymbol{\eta})$, $\tilde{y}_l(\boldsymbol{\eta},\alpha)$, $\tilde{T}(\boldsymbol{\eta},\alpha)$, and variables $p(\alpha|\boldsymbol{\eta})$, for all $\boldsymbol{\eta} \in \Omega$ and $\alpha \in \mathcal{A}$. Linear fractional programs are non-convex, although offline algorithms for solving them are known. For example, [8] provides a non-linear change of variables to turn the non-convex problem into a convex one. However, offline solutions to the above are not practical for two reasons. First, they would require full knowledge of the probability distribution $q(\boldsymbol{\eta})$, specifying a number of probabilities that is typically exponential in the dimension of the vector $\boldsymbol{\eta}$. Second, even if $q(\boldsymbol{\eta})$ were fully known, the problem would involve a number of variables $p(\alpha|\boldsymbol{\eta})$ that is huge, typically exponential in the dimension of $\boldsymbol{\eta}$.

The algorithms developed in this paper overcome the above challenges. They can be viewed as *online* algorithms for solving the linear fractional program. They make decisions over time that converge to optimality, without a-priori knowledge of the $q(\boldsymbol{\eta})$ probabilities, and without suffering from an exponential complexity or convergence time explosion when $\boldsymbol{\eta}$ has large dimension. Further, they allow for possibly uncountably infinite sets Ω and $\mathcal{A}$.

IV. The Dynamic Algorithm

This section develops an algorithm to solve the problem (9)-(11) to within any degree of accuracy.

A. Virtual Queues

To treat the constraints $\overline{y}_l/\overline{T} \leq c_l$, which are equivalent to the constraints $\overline{y}_l \leq c_l\overline{T}$, define *virtual queues* $Z_l[r]$ for $l \in \{1,\ldots,L\}$. The queues have initial condition $Z_l[0] = 0$ and update equation:

$$Z_l[r+1] = \max[Z_l[r]+y_l[r]-c_lT[r], 0] \forall l \in \{1,\ldots,L\} \quad (14)$$

The intuition is that if we *stabilize* the queue $Z_l[r]$, then the average of the "arrival process" $y_l[r]$ is less than or equal to that of the "service process" $c_lT[r]$, and so $\overline{y}_l \leq c_l\overline{T}$.

Lemma 2: If $Z_l[0] = 0$ and $Z_l[r]$ satisfies (14) for all $r \in \{0,1,2,\ldots\}$, then for all integers $R > 0$:

$$\tfrac{1}{R}\textstyle\sum_{r=0}^{R-1} y_l[r] - c_l\tfrac{1}{R}\sum_{r=0}^{R-1} T[r] \leq Z_l[R]/R \quad (15)$$

and hence:

$$\overline{y}_l[R] - c_l\overline{T}[R] \leq \mathbb{E}\left[Z_l[R]\right]/R \quad (16)$$

Proof: From (14) we have:

$$Z_l[r+1] \geq Z_l[r] + y_l[r] - c_lT[r]$$

Summing over $r \in \{0,1,\ldots,R-1\}$ gives:

$$Z_l[R] - Z_l[0] \geq \textstyle\sum_{r=0}^{R-1} y_l[r] - c_l\sum_{r=0}^{R-1} T[r]$$

Dividing by R and using $Z_l[0] = 0$ proves (15). Taking expectations proves (16). ■

It follows that if $\lim_{R\to\infty}\mathbb{E}\left[Z_l[R]\right]/R = 0$ for all $l \in \{1,\ldots,L\}$ (called *mean rate stability* [23]), then all desired constraints are satisfied.

B. Lyapunov Drift

Let $\boldsymbol{Z}[r] = (Z_1[r],\ldots,Z_L[r])$ be the vector of virtual queues, and define the following *Lyapunov function* $L[r]$:

$$L[r] \triangleq \tfrac{1}{2}\textstyle\sum_{l=1}^{L} Z_l[r]^2$$

The value $L[r]$ is a scalar measure of the size of the queue backlogs on frame r. The intuition is that if we take actions to consistently push this value down, then the queues will be stabilized. Define $\Delta[r]$ as the drift in the Lyapunov function from one frame to the next:

$$\Delta[r] \triangleq L[r+1] - L[r]$$

Lemma 3: Under any control decision for choosing $\pi[r] \in \mathcal{P}$, we have for all r and all possible $\boldsymbol{Z}[r]$:

$$\mathbb{E}\left[\Delta[r]|\boldsymbol{Z}[r]\right] \leq B + \mathbb{E}\left[\sum_{l=1}^{L} Z_l[r](y_l[r] - c_lT[r])|\boldsymbol{Z}[r]\right] \quad (17)$$

where B is a constant that satisfies for all r and all possible $\boldsymbol{Z}[r]$:

$$B \geq \frac{1}{2}\sum_{l=1}^{L}\mathbb{E}\left[(y_l[r] - c_lT[r])^2|\boldsymbol{Z}[r]\right] \quad (18)$$

Such a constant B exists by the boundedness assumptions (2)-(4). In particular, we can use:

$$B = \tfrac{dL}{2} + \tfrac{d}{2}\textstyle\sum_{l=1}^{L}(2|c_l| + c_l^2)$$

where d is defined in (4), although the sum term $(2|c_l| + c_l^2)$ can be reduced to c_l^2 if $2y_l[r]c_lT[r] \geq 0$ for all r.

Proof: Squaring (14) and using $\max[x,0]^2 \leq x^2$ yields:

$$\begin{aligned} Z_l[r+1]^2 &\leq (Z_l[r] + y_l[r] - c_lT[r])^2 \\ &= Z_l[r]^2 + (y_l[r] - c_lT[r])^2 \\ &\quad +2Z_l[r](y_l[r] - c_lT[r]) \end{aligned}$$

Taking conditional expectations, dividing by 2, and summing over $l \in \{1,\ldots,L\}$ yields the result. ■

C. The Drift-Plus-Penalty Ratio Algorithm

Our *Drift-Plus-Penalty Ratio Algorithm* is designed to minimize a sum of the variables on the right-hand-side of the drift bound (17) and a penalty term, divided by an expected frame size, as in [18]. The penalty term uses a non-negative constant V that will be shown to affect a performance tradeoff:

- (Policy Selection) Every frame $r \in \{0, 1, 2, \ldots\}$, observe the virtual queues $\boldsymbol{Z}[r]$ and choose a policy $\pi[r] \in \mathcal{P}$ to minimize the following expression:

$$\frac{\mathbb{E}\left[V\hat{y}_0(\pi[r]) + \sum_{l=1}^{L} Z_l[r]\hat{y}_l(\pi[r])|\boldsymbol{Z}[r]\right]}{\mathbb{E}\left[\hat{T}(\pi[r])|\boldsymbol{Z}[r]\right]} \tag{19}$$

- (Queue Update) Observe the resulting $\boldsymbol{y}[r]$ and $T[r]$ values, and update virtual queues $Z_l[r]$ by (14).

Details on minimizing (19) are given in Section V. Rather than assuming we achieve the exact infimum of (19) over all policies $\pi[r] \in \mathcal{P}$, it is useful to allow decisions to come within an additive constant C of the infimum.

Definition 1: A policy $\pi[r]$ is a *C-additive approximation* for the problem (19) if for a given constant $C \geq 0$ we have:

$$\frac{\mathbb{E}\left[V\hat{y}_0(\pi[r]) + \sum_{l=1}^{L} Z_l[r]\hat{y}_l(\pi[r])|\boldsymbol{Z}[r]\right]}{\mathbb{E}\left[\hat{T}(\pi[r])|\boldsymbol{Z}[r]\right]} \leq$$
$$C + \inf_{\pi\in\mathcal{P}}\left[\frac{\mathbb{E}\left[V\hat{y}_0(\pi) + \sum_{l=1}^{L} Z_l[r]\hat{y}_l(\pi)|\boldsymbol{Z}[r]\right]}{\mathbb{E}\left[\hat{T}(\pi)|\boldsymbol{Z}[r]\right]}\right]$$

Theorem 1: (Algorithm Performance) Assume the constraints of problem (9)-(11) are feasible. Fix constants $C \geq 0$, $V \geq 0$, and assume the above algorithm is implemented using any C-additive approximation every frame r for the minimization in (19). Then:

a) For all $l \in \{1, \ldots, L\}$ we have:

$$\limsup_{R\to\infty} \overline{y}_l[R]/\overline{T}[R] \leq c_l \tag{20}$$

$$\limsup_{R\to\infty} \frac{\sum_{r=0}^{R-1} y_l[r]}{\sum_{r=0}^{R-1} T[r]} \leq c_l \quad (w.p.1) \tag{21}$$

where $\overline{y}_l[R]$ and $\overline{T}[R]$ are defined in (5), and "w.p.1" stands for "with probability 1."

b) For all integers $R > 0$ we have:

$$\frac{\overline{y}_0[R]}{\overline{T}[R]} \leq ratio^{opt} + \frac{(B/\overline{T}[R] + C)}{V} \tag{22}$$

and hence:

$$\limsup_{R\to\infty} \overline{y}_0[R]/\overline{T}[R] \leq ratio^{opt} + (B/T^{min} + C)/V \tag{23}$$

where B is defined in (18), and $ratio^{opt}$ is the optimal solution to (9)-(11).

c) There are constants F_1, F_2 such that for all $R > 0$:

$$\frac{\mathbb{E}\left[||\boldsymbol{Z}[R]||\right]}{R} \leq \sqrt{\frac{F_1 + VF_2}{R}}$$

where $||\boldsymbol{Z}[R]|| = \sqrt{\sum_{l=1}^{L} Z_l[R]^2}$ is the Euclidean norm, and is at least as large as any component $Z_l[R]$.

Thus, the algorithm satisfies all constraints, and the value of V can be chosen appropriately large to make $(B/T^{min} + C)/V$ arbitrarily small, ensuring that the time average penalty is arbitrarily close to its optimal value $ratio^{opt}$. The tradeoff in choosing a large value of V comes in the size of the $Z_l[r]$ queues and the number of frames R required for $\mathbb{E}\left[Z_l[R]\right]/R$ to approach zero, which affects convergence time to the desired constraints. In particular, by Lemma 2 we have:

$$\overline{y}_l[R] - c_l\overline{T}[R] \leq \sqrt{(F_1 + VF_2)/R} \tag{24}$$

It is clear that the right-hand-side above vanishes as $R \to \infty$, but the number of frames required for it to be small depends on the V parameter. Defining $\epsilon = 1/V$ shows by (22) and (24) that the time average objective is within $O(\epsilon)$ from optimality, with constraint violations that decay with R like $\sqrt{1/(\epsilon R)}$. Under a mild "Slater-type" assumption that ensures the constraints (10) are achievable with "δ-slackness" for some $\delta > 0$, the queues $Z_l[R]$ can be shown to be *strongly stable*, in the sense that the time average expectation is bounded by $O(V)$, which leads to a tighter bound than (24) (see related analysis in [23]). The algorithm in that case gives $O(\epsilon)$ distance to the optimal objective with constraint violations that decay like $1/(\epsilon R)$, rather than $\sqrt{1/(\epsilon R)}$.

The constants F_1, F_2 are exactly computed in the proof of Theorem 1, and hence one can choose the V parameter according to the bounds (24) and (23). However, these bounds are often conservative and one can typically achieve desired performance with smaller values of V than those indicated by these bounds. Simulations that demonstrate the tradeoff with V are provided in Section VII-C.

D. Proof of Theorem 1

We first present a lemma about minimizing a ratio of expectations of the form $\mathbb{E}\left[a(\pi)\right]/\mathbb{E}\left[b(\pi)\right]$. Let $\mathcal{P}$ be an abstract (possibly uncountably infinite) set, and let $a(\pi)$, $b(\pi)$ be random functions of $\pi \in \mathcal{P}$. Assume expectations $\mathbb{E}\left[a(\pi)\right]$ and $\mathbb{E}\left[b(\pi)\right]$ are well defined for all $\pi \in \mathcal{P}$, and that there is a constant $b_{min} > 0$ such that $\mathbb{E}\left[b(\pi)\right] \geq b_{min}$ for all $\pi \in \mathcal{P}$. Define $\mathcal{F}$ as the set of all vectors $\left(\mathbb{E}\left[a(\pi)\right], \mathbb{E}\left[b(\pi)\right]\right)$ for all $\pi \in \mathcal{P}$, and assume this set is bounded.

Now let π^* denote a *mixed policy*, which randomly chooses a policy in $\mathcal{P}$ with some probability distribution, so that the expectations $\mathbb{E}\left[a(\pi^*)\right]$, $\mathbb{E}\left[b(\pi^*)\right]$ are with respect to the randomness of π^* and the randomness of the functions $a(\cdot)$ and $b(\cdot)$ given the particular π^* chosen. The next lemma shows that mixed strategies cannot help to reduce the ratio beyond the infimum attainable over non-mixed policies.

Lemma 4: Let $a(\pi)$ and $b(\pi)$ be defined as above. Then for any mixed policy π^* we have:

$$\frac{\mathbb{E}\left[a(\pi^*)\right]}{\mathbb{E}\left[b(\pi^*)\right]} \geq \inf_{\pi\in\mathcal{P}}\left[\frac{\mathbb{E}\left[a(\pi)\right]}{\mathbb{E}\left[b(\pi)\right]}\right]$$

Proof: (Lemma 4) Let π^* be any mixed policy. Given a particular outcome $\pi^* = \pi$, the expectation $\mathbb{E}\left[(a(\pi^*), b(\pi^*))|\pi^* = \pi\right]$ is in the set $\mathcal{F}$, and so the unconditional expectation $\mathbb{E}\left[(a(\pi^*), b(\pi^*))\right]$ is in the convex hull of $\mathcal{F}$. Thus, it can be achieved as a convex combination of

points in $\mathcal{F}$. That is, we can find a finite integer M, policies $\pi_1, \ldots, \pi_M$ in $\mathcal{P}$, and probabilities $q_1, \ldots, q_M$ such that:

$$(\mathbb{E}\left[a(\pi^*)\right], \mathbb{E}\left[b(\pi^*)\right]) = \sum_{i=1}^{M} q_i (\mathbb{E}\left[a(\pi_i)\right], \mathbb{E}\left[b(\pi_i)\right])$$

Recall that $\mathbb{E}\left[b(\pi_i)\right] > 0$ for all $i \in \{1, \ldots, M\}$. Thus:

$$\begin{aligned}
\frac{\mathbb{E}\left[a(\pi^*)\right]}{\mathbb{E}\left[b(\pi^*)\right]} &= \frac{\sum_{i=1}^{M} q_i \mathbb{E}\left[a(\pi_i)\right]}{\sum_{i=1}^{M} q_i \mathbb{E}\left[b(\pi_i)\right]} \\
&= \frac{\sum_{i=1}^{M} q_i \mathbb{E}\left[b(\pi_i)\right] \frac{\mathbb{E}[a(\pi_i)]}{\mathbb{E}[b(\pi_i)]}}{\sum_{i=1}^{M} q_i \mathbb{E}\left[b(\pi_i)\right]} \\
&\geq \frac{\sum_{i=1}^{M} q_i \mathbb{E}\left[b(\pi_i)\right] \inf_{\pi \in \mathcal{P}} \left[\frac{\mathbb{E}[a(\pi)]}{\mathbb{E}[b(\pi)]}\right]}{\sum_{i=1}^{M} q_i \mathbb{E}\left[b(\pi_i)\right]} \\
&= \inf_{\pi \in \mathcal{P}} \left[\frac{\mathbb{E}\left[a(\pi)\right]}{\mathbb{E}\left[b(\pi)\right]}\right]
\end{aligned}$$

■

The above lemma shows that the infimum of the ratio of expectations in (19) over all policies $\pi[r] \in \mathcal{P}$ is less than or equal to the corresponding ratio achieved by any i.i.d. algorithm that *randomly* selects $\pi^*[r] \in \mathcal{P}$ according to some probability distribution. Thus, if policy $\pi[r]$ is a C-additive approximation, then:

$$\frac{\mathbb{E}\left[V\hat{y}_0(\pi[r]) + \sum_{l=1}^{L} Z_l[r]\hat{y}_l(\pi[r]) | \boldsymbol{Z}[r]\right]}{\mathbb{E}\left[\hat{T}(\pi[r]) | \boldsymbol{Z}[r]\right]} \leq C + \frac{V\mathbb{E}\left[\hat{y}_0(\pi^*[r])\right] + \sum_{l=1}^{L} Z_l[r]\mathbb{E}\left[\hat{y}_l(\pi^*[r])\right]}{\mathbb{E}\left[\hat{T}(\pi^*[r])\right]} \quad (25)$$

where $\pi^*[r]$ is any i.i.d. algorithm. Note that under an i.i.d. algorithm $\pi^*[r]$, the conditional expectations of $\hat{y}_l(\pi^*[r])$ and $\hat{T}(\pi^*[r])$, given $\boldsymbol{Z}[r]$, are the same as unconditional expectations. This is because their decisions are independent of system history. We now prove Theorem 1.

Proof: (Theorem 1) We first prove part (b). Fix $r \in \{0, 1, 2, \ldots\}$. Adding $\mathbb{E}\left[V\hat{y}_0(\pi[r]) | \boldsymbol{Z}[r]\right]$ to (17) gives:

$$\begin{aligned}
&\mathbb{E}\left[\Delta[r] + V\hat{y}_0(\pi[r]) | \boldsymbol{Z}[r]\right] \leq B \\
&-\mathbb{E}\left[\hat{T}(\pi[r]) | \boldsymbol{Z}[r]\right] \sum_{l=1}^{L} Z_l[r]c_l \\
&+\mathbb{E}\left[V\hat{y}_0(\pi[r]) + \sum_{l=1}^{L} Z_l[r]\hat{y}_l(\pi[r]) | \boldsymbol{Z}[r]\right]
\end{aligned}$$

Substituting (25) into the last term above yields:

$$\begin{aligned}
&\mathbb{E}\left[\Delta[r] + V\hat{y}_0(\pi[r]) | \boldsymbol{Z}[r]\right] \leq B \\
&-\mathbb{E}\left[\hat{T}(\pi[r]) | \boldsymbol{Z}[r]\right] \sum_{l=1}^{L} Z_l[r]c_l \\
&+\mathbb{E}\left[\hat{T}(\pi[r]) | \boldsymbol{Z}[r]\right] \left[C + \frac{V\mathbb{E}[\hat{y}_0(\pi^*[r])] + \sum_{l=1}^{L} Z_l[r]\mathbb{E}[\hat{y}_l(\pi^*[r])]}{\mathbb{E}\left[\hat{T}(\pi^*[r])\right]}\right]
\end{aligned}$$

In the above inequality, $\pi[r]$ represents the C-additive approximate decision actually made, and $\pi^*[r]$ is from any alternative i.i.d. algorithm. Now recall that for any $\delta > 0$, (12)-(13) imply the existence of an i.i.d. algorithm $\pi^*[r]$ that satisfies:

$$\begin{aligned}
\mathbb{E}\left[\hat{y}_0(\pi^*[r])\right] / \mathbb{E}\left[\hat{T}(\pi^*[r])\right] &\leq ratio^{opt} + \delta \\
\mathbb{E}\left[\hat{y}_l(\pi^*[r])\right] / \mathbb{E}\left[\hat{T}(\pi^*[r])\right] &\leq c_l + \delta \quad \forall l \in \{1, \ldots, L\}
\end{aligned}$$

Substituting these into the right-hand-side of the previous drift expression yields:

$$\begin{aligned}
&\mathbb{E}\left[\Delta[r] + V\hat{y}_0(\pi[r]) | \boldsymbol{Z}[r]\right] \leq B \\
&+\mathbb{E}\left[\hat{T}(\pi[r]) | \boldsymbol{Z}[r]\right] \left[C + V(ratio^{opt} + \delta) + \sum_{l=1}^{L} Z_l[r]\delta\right]
\end{aligned}$$

The above holds for all $\delta > 0$. Taking a limit as $\delta \to 0$ in the above yields:

$$\begin{aligned}
&\mathbb{E}\left[\Delta[r] + V\hat{y}_0(\pi[r]) | \boldsymbol{Z}[r]\right] \leq B \\
&+\mathbb{E}\left[\hat{T}(\pi[r]) | \boldsymbol{Z}[r]\right] \left[C + Vratio^{opt}\right] \quad (26)
\end{aligned}$$

Taking expectations of the above yields:

$$\begin{aligned}
&\mathbb{E}\left[L[r+1]\right] - \mathbb{E}\left[L[r]\right] + V\mathbb{E}\left[\hat{y}_0(\pi[r])\right] \leq \\
&B + \mathbb{E}\left[\hat{T}(\pi[r])\right] \left[C + Vratio^{opt}\right]
\end{aligned}$$

Summing the above over $r \in \{0, \ldots, R-1\}$ for some integer $R > 0$ and dividing by R yields:

$$\frac{\mathbb{E}\left[L[R]\right] - \mathbb{E}\left[L[0]\right]}{R} + V\overline{y}_0[R] \leq B + \overline{T}[R][C + Vratio^{opt}]$$

Rearranging terms in the above and using $\mathbb{E}\left[L[R]\right] \geq 0$, $\mathbb{E}\left[L[0]\right] = 0$ yields the result of part (b).

To prove part (c), taking expectations of (26) gives:

$$\mathbb{E}\left[\Delta[r]\right] + V\mathbb{E}\left[y_0[r]\right] \leq B + \mathbb{E}\left[T[r]\right](C + Vratio^{opt})$$

and so for all $r \in \{0, 1, 2, \ldots\}$:

$$\mathbb{E}\left[\Delta[r]\right] \leq A_1 + VA_2 \quad (27)$$

where A_1, A_2 are constants that satisfy for all r:

$$\begin{aligned}
A_1 &\geq B + C\mathbb{E}\left[T[r]\right] \\
A_2 &\geq ratio^{opt}\mathbb{E}\left[T[r]\right] - \mathbb{E}\left[y_0[r]\right]
\end{aligned}$$

Summing (27) over $r \in \{0, \ldots, R-1\}$ gives:

$$\mathbb{E}\left[L[R]\right] - \mathbb{E}\left[L[0]\right] \leq (A_1 + VA_2)R$$

Using $\mathbb{E}\left[L[0]\right] = 0$ and $L[R] = \frac{1}{2}||\boldsymbol{Z}[R]||^2$ gives:

$$\mathbb{E}\left[||\boldsymbol{Z}[R]||^2\right] \leq 2(A_1 + VA_2)R$$

Clearly $\mathbb{E}\left[X\right]^2 \leq \mathbb{E}\left[X^2\right]$ for any random variable X, and so:

$$\mathbb{E}\left[||\boldsymbol{Z}[R]||\right]^2 \leq 2(A_1 + VA_2)R$$

Dividing by R^2 and taking a square root yields:

$$\mathbb{E}\left[||\boldsymbol{Z}[R]||\right]/R \leq \sqrt{(2A_1 + 2VA_2)/R}$$

Part (c) follows by defining $F_1 \triangleq 2A_1$, $F_2 \triangleq 2A_2$.

To prove part (a), note that (27) implies that for all r we have $\mathbb{E}\left[\Delta[r]\right] \leq F$ for some constant F. Further, the second moments of per-frame changes in $Z_l[r]$ are bounded because of the second moment assumptions on $y_l[r]$ and $T[r]$. It follows that (see [25]):

$$\begin{aligned}
\lim_{R\to\infty} \mathbb{E}\left[Z_l[R]\right]/R &= 0 \quad (28) \\
\lim_{R\to\infty} Z_l[R]/R &= 0 \quad (w.p.1) \quad (29)
\end{aligned}$$

Part (a) then follows from Lemma 2. ■

V. Optimizing the Ratio of Expectations

This section shows how to minimize the ratio of expectations in (19), which is the crucial step of the ratio algorithm every frame r. Such a problem can be written more simply as choosing a policy $\pi \in \mathcal{P}$ to minimize the ratio:

$$\frac{\mathbb{E}\left[a(\pi)\right]}{\mathbb{E}\left[b(\pi)\right]}$$

where $a(\pi), b(\pi)$ are random functions of $\pi \in \mathcal{P}$, with expectations that are bounded by finite constants independent of $\pi \in \mathcal{P}$. The function $b(\pi)$ is equal to $\hat{T}(\pi)$, and is strictly positive and satisfies the following for all $\pi \in \mathcal{P}$:

$$0 < T^{min} \leq \mathbb{E}\left[b(\pi)|\pi\right] \leq T^{max} < \infty$$

The function $a(\pi)$ depends on $\boldsymbol{Z}[r]$ and is defined as the expression inside the expectation of the numerator in (19). The expectations $\mathbb{E}\left[a(\pi)\right]$ and $\mathbb{E}\left[b(\pi)\right]$ are implicitly conditioned on $\boldsymbol{Z}[r]$, although this notation is suppressed in this section for simplicity.

First consider the special case of the protocol selection model of Section II-B, where the policy space $\mathcal{P}$ contains a finite collection of M policy options for which the expectations $\mathbb{E}\left[a(\pi)\right]$ and $\mathbb{E}\left[b(\pi)\right]$ are known (possibly via some accurate estimation as discussed in Section II-B). By Lemma 4, the minimizing ratio is obtained simply by testing all M options and choosing the option π that minimizes $\mathbb{E}\left[a(\pi)\right]/\mathbb{E}\left[b(\pi)\right]$.

Next, consider another special case: Assume the policy space $\mathcal{P}$ is infinite. However, assume $\mathbb{E}\left[b(\pi)\right]$ is the same for all $\pi \in \mathcal{P}$. Then minimizing the ratio reduces to the easier problem of minimizing $\mathbb{E}\left[a(\pi)\right]$.

General problems that have infinite policy spaces with $\mathbb{E}\left[b(\pi)\right]$ that depends on π are more challenging. The remainder of this section is devoted to such problems.

A. Reduction to a Single Expectation

Define θ^* as the optimal ratio:

$$\theta^* \triangleq \inf_{\pi\in\mathcal{P}} \left[\frac{\mathbb{E}\left[a(\pi)\right]}{\mathbb{E}\left[b(\pi)\right]}\right]$$

Lemma 5: We have:

$$\inf_{\pi\in\mathcal{P}} \mathbb{E}\left[a(\pi) - \theta^* b(\pi)\right] = 0 \tag{30}$$

Further, for any real number θ, we have:

$$\inf_{\pi\in\mathcal{P}} \mathbb{E}\left[a(\pi) - \theta b(\pi)\right] < 0 \quad \text{if } \theta > \theta^* \tag{31}$$

$$\inf_{\pi\in\mathcal{P}} \mathbb{E}\left[a(\pi) - \theta b(\pi)\right] > 0 \quad \text{if } \theta < \theta^* \tag{32}$$

Proof: By definition of θ^*, we have $\mathbb{E}\left[a(\pi)\right]/\mathbb{E}\left[b(\pi)\right] \geq \theta^*$ for any policy $\pi \in \mathcal{P}$. Thus:

$$\mathbb{E}\left[a(\pi) - \theta^* b(\pi)\right] \geq 0$$

This holds for any $\pi \in \mathcal{P}$, and so taking the infimum:

$$\inf_{\pi\in\mathcal{P}} \mathbb{E}\left[a(\pi) - \theta^* b(\pi)\right] \geq 0$$

However, it is easy to show that 0 can be approached arbitrarily closely using policies π with ratios $\mathbb{E}\left[a(\pi)\right]/\mathbb{E}\left[b(\pi)\right]$ that approach θ^* arbitrarily closely. This proves (30).

We now use (30) to prove (31)-(32). Suppose that $\theta > \theta^*$. We then have for any $\pi \in \mathcal{P}$:

$$\begin{aligned}\mathbb{E}\left[a(\pi) - \theta b(\pi)\right] &= \mathbb{E}\left[a(\pi) - \theta^* b(\pi) - (\theta - \theta^*)b(\pi)\right] \\ &\leq \mathbb{E}\left[a(\pi) - \theta^* b(\pi)\right] - (\theta - \theta^*)T^{min}\end{aligned}$$

Thus:

$$\begin{aligned}\inf_{\pi\in\mathcal{P}} \mathbb{E}\left[a(\pi) - \theta b(\pi)\right] &\leq \inf_{\pi\in\mathcal{P}} \mathbb{E}\left[a(\pi) - \theta^* b(\pi)\right] \\ &\quad -(\theta - \theta^*)T^{min} \\ &= 0 - (\theta - \theta^*)T^{min} < 0\end{aligned}$$

where the equality holds by (30). This proves (31). Inequality (32) follows by a similar argument. ∎

The above lemma shows that a specific policy $\tilde{\pi}$ achieves the infimum in (30) if and only if $\mathbb{E}\left[a(\tilde{\pi})\right]/\mathbb{E}\left[b(\tilde{\pi})\right] = \theta^*$, and that $\theta = \theta^*$ if and only if $\inf_{\pi\in\mathcal{P}} \mathbb{E}\left[a(\pi) - \theta b(\pi)\right] = 0$.

B. The Bisection Algorithm

Lemma 5 immediately leads to the following simple bisection algorithm: Suppose we have upper and lower bounds θ_{min} and θ_{max}, so that we know $\theta_{min} \leq \theta^* \leq \theta_{max}$. Then we can define $\theta = (\theta_{min} + \theta_{max})/2$, and compute the value of $\inf_{\pi\in\mathcal{P}} \mathbb{E}\left[a(\pi) - \theta b(\pi)\right]$. If the result is 0, then $\theta = \theta^*$. If positive, then $\theta < \theta^*$, and otherwise $\theta > \theta^*$. We can then refine our upper and lower bounds. This leads to a simple iterative algorithm where the distance between the upper and lower bounds decreases by a factor of 2 on each iteration. It thus approaches the optimal θ^* value exponentially fast. Each step of the iteration involves minimizing an expectation, rather than a ratio of expectations.

C. Optimizing for the Single Decision Model

Consider the single decision policies of Section II-A, which observe $\boldsymbol{\eta}[r]$ every frame r and choose $\alpha[r] \in \mathcal{A}_{\boldsymbol{\eta}[r]}$ to minimize:

$$\frac{\mathbb{E}\left[\tilde{a}(\boldsymbol{\eta}[r], \alpha[r])\right]}{\mathbb{E}\left[\tilde{b}(\boldsymbol{\eta}[r], \alpha[r])\right]}$$

where $\tilde{a}(\boldsymbol{\eta}, \alpha)$ and $\tilde{b}(\boldsymbol{\eta}, \alpha)$ for frame r are deterministic functions given by the numerator and denominator in (19):

$$\tilde{b}(\boldsymbol{\eta}, \alpha) = \tilde{T}(\boldsymbol{\eta}, \alpha)\ ,\ \tilde{a}(\boldsymbol{\eta}, \alpha) = V\tilde{y}_0(\boldsymbol{\eta}, \alpha) + \sum_{l=1}^{L} Z_l[r]\tilde{y}_l(\boldsymbol{\eta}, \alpha)$$

Unlike minimizing a single expectation, the minimum for this ratio of expectations is generally *not* achieved by observing $\boldsymbol{\eta}[r]$ and choosing $\alpha \in \mathcal{A}_{\boldsymbol{\eta}[r]}$ to greedily minimize the deterministic function $\tilde{a}(\boldsymbol{\eta}[r], \alpha)/\tilde{b}(\boldsymbol{\eta}[r], \alpha)$.

A correct approach is the following: If θ^* is known, simply choose $\alpha \in \mathcal{A}_{\boldsymbol{\eta}[r]}$ to greedily minimize the following deterministic function:

$$\tilde{a}(\boldsymbol{\eta}[r], \alpha) - \theta^*\tilde{b}(\boldsymbol{\eta}[r], \alpha)$$

This ensures that $\mathbb{E}\left[\tilde{a}(\boldsymbol{\eta}[r], \alpha[r]) - \theta^*\tilde{b}(\boldsymbol{\eta}[r], \alpha[r])\right]$ is minimized,[3] which by Lemma 5 ensures our action achieves the

[3]Formally, this is by the principle of opportunistically minimizing an expectation, see Section 1.7 of [23].

optimal ratio of expectations θ^*. If θ^* is unknown, we can carry out the bisection routine. Let θ be the midpoint in the current iteration. We must compute:

$$\inf_{\pi\in\mathcal{P}} \mathbb{E}\left[a(\pi)-\theta b(\pi)\right] = \mathbb{E}\left[\inf_{\alpha\in\mathcal{A}_{\boldsymbol{\eta}}}\left[\tilde{a}(\boldsymbol{\eta},\alpha)-\theta\tilde{b}(\boldsymbol{\eta},\alpha)\right]\right] \quad (33)$$

where the expectation on the right-hand-side is with respect to the distribution of $\boldsymbol{\eta}$. The infimizing decision can be made by observing $\boldsymbol{\eta}$, and choosing $\alpha\in\mathcal{A}_{\boldsymbol{\eta}}$ to greedily minimize:

$$\tilde{a}(\boldsymbol{\eta},\alpha)-\theta\tilde{b}(\boldsymbol{\eta},\alpha)$$

This does not require knowledge of the $\boldsymbol{\eta}$ distribution. However, the value in (33) cannot be computed without knowledge of this distribution, and the bisection routine requires knowing whether or not this value is positive. To obtain an approximation for this value, suppose we have W i.i.d. samples $\{\boldsymbol{\eta}_w\}_{w=1}^W$. We can then approximate the value in (33) by the function $val(\theta)$ defined below:

$$val(\theta)\triangleq\frac{1}{W}\sum_{w=1}^W \inf_{\alpha\in\mathcal{A}_{\boldsymbol{\eta}_w}}\left[\tilde{a}(\alpha,\boldsymbol{\eta}_w)-\theta\tilde{b}(\alpha,\boldsymbol{\eta}_w)\right] \quad (34)$$

By the law of large numbers, $val(\theta)$ approaches the exact value of (33) with a large choice of W. Indeed, let $val_w(\theta)$ be the infimum in each term of the above sum, $val(\theta)$ be the approximation (34) obtained by averaging the $val_w(\theta)$ values, and $val^*(\theta)$ be the exact value in (33). Then the mean-square error decays with W:

$$\begin{aligned}&\mathbb{E}\left[(val(\theta)-val^*(\theta))^2\right]\\ &=\mathbb{E}\left[\left(\frac{1}{W}\sum_{w=1}^W(val_w(\theta)-val^*(\theta))\right)^2\right]=\sigma^2/W\end{aligned}$$

where σ^2 is the variance of $val_1(\theta)$. The bisection routine can be carried out using the $val(\theta)$ approximation, being sure to use the same samples (and hence the same $val(\theta)$ function) at each step of the iteration procedure for a given frame, but updating samples at the start of each new frame. Note that $val(\theta)$ in (34) is non-increasing in θ, so the bisection procedure carried out on each frame r will converge, provided that it is initialized so that $val(\theta_{min})\geq 0$ and $val(\theta_{max})\leq 0$. If we cannot independently generate W samples, we use the W past observed values of $\boldsymbol{\eta}[r]$ from previous frames. There is a subtle issue here, as these past values have influenced system performance and are thus correlated with the current $a(\pi)$ and $b(\pi)$ functions. However, a *delayed queue argument* similar to that given in [26] shows these past values can still be used. A simulation example in Section VII-C shows that the bisection algorithm typically needs only 10-12 iterations for a close minimization of $val(\theta)$ over $\theta\in[\theta_{min},\theta_{max}]$. Further, the number of samples W does not need to be large for overall performance to be close to optimal. This is because we only need to determine if (33) is positive or negative, which only requires accurate estimation when it is close to 0, which is exactly when θ is already close to θ^*.

D. MDP models

Consider now a discrete time MDP model, as described in Section II-C. Assume the state space is finite, and that there is a single state 0 that is recurrent with bounded first and second moments of recurrence time and penalties under any policy. We have two methods for minimizing the ratio of expectations. The first carries out the bisection routine with fixed values of θ at each iteration, where we want to minimize $\mathbb{E}\left[a(\pi)-\theta b(\pi)\right]$. For simplicity of notation, we concentrate on a single frame r, fix $Z_l[r]=Z_l$, and suppress the notation for conditional expectation given $\boldsymbol{Z}[r]$ (so this conditioning is implicitly understood). Using the definition of $b(\pi)=\hat{T}(\pi)$, and $a(\pi)$ equal to the function inside the expectation of the numerator in (19), this is equivalent to minimizing the expectation of:

$$V\hat{y}_0(\pi)+\sum_{l=1}^L Z_l\hat{y}_l(\pi)-\theta\hat{T}(\pi)$$

The penalties $\hat{y}_l(\pi)$ and frame size $\hat{T}(\pi)$ can be written as sums of penalties incurred over each slot τ of the frame:

$$\hat{y}_l(\pi)=\sum_{\tau=0}^{\hat{T}(\pi)-1}\hat{y}_l(\pi,\tau)\ ,\ \hat{T}_l(\pi)=\sum_{\tau=0}^{\hat{T}(\pi)-1}1$$

where $\hat{y}_l(\pi,\tau)$ is the lth penalty incurred by policy π at slot τ of the renewal frame. The expectation to minimize can then be written as a general *stochastic shortest path problem* of minimizing the expectation of:

$$\sum_{\tau=0}^{\hat{T}(\pi)-1}\left[V\hat{y}_0(\pi,\tau)+\sum_{l=1}^L Z_l\hat{y}_l(\pi,\tau)-\theta\right]$$

Such stochastic shortest path problems can be solved using the neuro-dynamic programming methods in [7].

The second approach uses neither bisection nor a θ variable. It is based on the following observation from renewal theory. The policy that minimizes the ratio of expectations in (19) is identical to the policy that minimizes the *infinite horizon cost in a virtual unconstrained MDP problem*. The virtual unconstrained MDP problem is defined for fixed weights $Z_1,\ldots,Z_L$, and seeks to minimize the following infinite horizon time average cost:

$$\lim_{t\to\infty}\frac{1}{t}\sum_{\tau=0}^{t-1}\mathbb{E}\left[V\hat{y}_0(\pi,\tau)+\sum_{l=1}^L Z_l\hat{y}_l(\pi,\tau)\right] \quad (35)$$

This observation was made in [27] and [28], based on earlier versions of the current work [1]. Such infinite horizon average cost problems can be solved by value or policy iteration of a Bellman equation if all system probabilities are known [29].

The shortest path problem and the infinite average cost problem both suffer from a complexity explosion when the state space of the underlying MDP is large. However, recent low complexity, yet exact, applications of the method (35) are given in [27] for optimizing average delay in multi-class queueing systems, and in [28] for cognitive radio problems. Remarkably, the algorithms in [27][28] do not require probability knowledge and do not suffer in performance or convergence time because of their large state spaces.

It is important to emphasize that the infinite horizon problem (35) would need to be re-solved for different $Z_l[r]$ values on each frame. Fortunately, in "steady state" and for large V, the $Z_l[r]$ values do not vary significantly with respect to their average size, and so the policy computed on the previous frame is often an accurate initial seed in computing the policy for the new frame.

VI. Low Complexity Alternative Algorithms

A. Alternative Formulation

Consider constraints of the form $\overline{y}_l \leq 0$. These are equivalent to $\overline{y}_l/\overline{T} \leq c_l$ in the special case $c_l = 0$, and thus can be handled using the drift-plus-penalty ratio framework of Section IV. Now consider the following alternative problem:

$$\begin{aligned} \text{Minimize:} \quad & \overline{y}_0 \\ \text{Subject to:} \quad & \overline{y}_l/\overline{T} \leq c_l \ \forall l \in \{1,\ldots,L\} \\ & \pi[r] \in \mathcal{P} \ \forall r \in \{0,1,2,\ldots\} \end{aligned}$$

This has a different structure than the problem (6)-(8). However, it can be mapped to an instance of the original framework using the following equivalent problem:

$$\begin{aligned} \text{Minimize:} \quad & \overline{y}_0/\overline{1} \\ \text{Subject to:} \quad & (\overline{y}_l - c_l\overline{T})/\overline{1} \leq 0 \ \forall l \in \{1,\ldots,L\} \\ & \pi[r] \in \mathcal{P} \ \forall r \in \{0,1,2,\ldots\} \end{aligned}$$

where $\overline{1} = 1$, representing the average frame size in a system where all frames have size 1. This new problem defines frame sizes, penalties, and constants $\hat{T}_{new}(\pi)$, $\hat{y}_{l,new}(\pi)$, $c_{l,new}$ by:

$$\hat{T}_{new}(\pi) = 1 \ , \ \hat{y}_{l,new}(\pi) = \hat{y}_l(\pi) - c_l\hat{T}(\pi) \ , \ c_{l,new} = 0$$

Therefore, it can be solved with the drift-plus-penalty ratio algorithm. In this special case, the resulting algorithm uses virtual queues $Z_l[r]$ with updates:

$$Z_l[r+1] = \max[Z_l[r] + \hat{y}_l(\pi[r]) - c_l\hat{T}(\pi[r]), 0] \forall l \in \{1,\ldots,L\} \quad (36)$$

Every frame r, it observes the $Z_l[r]$ values and chooses $\pi[r] \in \mathcal{P}$ to minimize:

$$\mathbb{E}\left[V\hat{y}_0(\pi[r]) + \sum_{l=1}^{L} Z_l[r][\hat{y}_l(\pi[r]) - c_l\hat{T}(\pi[r])]|\boldsymbol{Z}[r]\right]$$

It then updates the virtual queues via (36). This procedure involves minimizing a single expectation. In the special case of the single decision model with initial information $\boldsymbol{\eta}[r]$, it reduces to a simple greedy action of observing $\boldsymbol{\eta}[r]$ every frame and choosing $\alpha[r] \in \mathcal{A}_{\boldsymbol{\eta}[r]}$ to deterministically minimize:

$$V\tilde{y}_0(\boldsymbol{\eta}[r], \alpha[r]) + \sum_{l=1}^{L} Z_l[r][\tilde{y}_l(\boldsymbol{\eta}[r], \alpha[r]) - c_l\tilde{T}(\boldsymbol{\eta}[r], \alpha[r])] \quad (37)$$

This does not require knowledge of the probability distribution for $\boldsymbol{\eta}[r]$, and does not require any bisection procedure.

B. Alternative Algorithm

The following is an alternative algorithm for the original problem (6)-(8) that does not require a ratio minimization (and hence does not require a bisection step): Use the same virtual queues $Z_l[r]$ in (14). Define $\theta[0] = 0$, and define $\theta[R]$ for $R \in \{1,2,3,\ldots\}$ by:

$$\theta[R] \triangleq \textstyle\sum_{r=0}^{R-1} y_0[r] / \sum_{r=0}^{R-1} T[r] \quad (38)$$

Every frame r, observe $\boldsymbol{Z}[r]$ and $\theta[r]$ and select a policy $\pi[r] \in \mathcal{P}$ to minimize the following expression:

$$\begin{aligned} &\mathbb{E}\left[V[\hat{y}_0(\pi[r]) - \theta[r]\hat{T}(\pi[r])]|\boldsymbol{Z}[r], \theta[r]\right] \quad (39) \\ &+\mathbb{E}\left[\textstyle\sum_{l=1}^{L} Z_l[r][\hat{y}_l(\pi[r]) - c_l\hat{T}(\pi[r])]|\boldsymbol{Z}[r], \theta[r]\right] \end{aligned}$$

It is shown in [23][24] that all constraints are met, and that if $\theta[r]$ converges to a constant with probability 1, then with probability 1:

$$\lim_{R\to\infty} \textstyle\sum_{r=0}^{R-1} y_0[r] / \sum_{r=0}^{R-1} T[r] \leq ratio^{opt} + O(1/V)$$

The disadvantage is that the convergence time is not as clear as that given in part (b) of Theorem 1. Further, use of the time average (38) makes it difficult to adapt to changes in system parameters, so that it may be better to approximate (38) with a moving average or an exponentially decaying average.

VII. Examples

This section presents example applications to a peer-to-peer wireless network, a transportation system, and a quality-aware task processing network. All of these examples use the structure of single decision policies with initial information, as discussed in Section II-A.

A. Wireless Peer-to-Peer File Downloads

Wireless device-to-device scheduling for peer-to-peer communication is a topic of recent interest (see, for example, [30][31] and references therein). Consider a wireless user that can receive peer-to-peer file downloads from L different wireless devices. It repeatedly requests downloads, one file after the other. A renewal is defined at the start of each new download. At the beginning of every renewal, the user desires a new file and sends a query to all L devices to ask if they have the file. Assume the query plus feedback incurs a fixed time T_{query}. The feedback on frame r specifies: (i) the subset $\mathcal{H}[r] \subseteq \{1,\ldots,L\}$ of devices that have the file, (ii) for each device $i \in \mathcal{H}[r]$, the time $T_{i,transmit}[r]$ and energy $e_{i,transmit}[r]$ that would be required to transmit the file over its channel. For simplicity, assume the channel states do not change during a frame, so that transmit times and energy use can be calculated based on the file size for frame r and the channel condition for each device i. This feedback information can be viewed as the vector $\boldsymbol{\eta}[r]$ of initial information. Assume that $\mathcal{H}[r]$ is non-empty for all r, so that there is always at least one device that contains the file. Every frame r, the wireless user must select a single device $i \in \mathcal{H}[r]$ from which to receive its download. That device alone expends energy $e_{i,transmit}[r]$ for frame r (energy due to querying is neglected for simplicity). The goal is to maximize the file download rate subject to average power constraints $P_{av,i}$ at each device $i \in \{1,\ldots,L\}$, where $P_{av,i}$ are given positive constants.

Define the following attributes:

$$\begin{aligned} energy_i[r] &\triangleq \text{energy used by device } i \text{ on frame } r. \\ T[r] &\triangleq T_{query} + \text{transmission time for frame } r. \end{aligned}$$

These attributes are deterministic functions of $\boldsymbol{\eta}[r]$ and the action chosen. The problem becomes:

$$\begin{aligned} &\text{Maximize:} && 1/\overline{T} \\ &\text{Subject to:} && \overline{energy}_i/\overline{T} \leq P_{av,i} \ \forall i \in \{1,\ldots,L\} \end{aligned}$$

This is equivalent to the following:

$$\begin{aligned} &\text{Minimize:} && \overline{T}/1 \\ &\text{Subject to:} && (\overline{energy}_i - P_{av,i}\overline{T})/1 \leq 0 \ \forall i \in \{1,\ldots,L\} \end{aligned}$$

This fits the special case framework of Section VI-A (with "effective" frame size of 1), and so the algorithm is given by (36)-(37): Define virtual queues $Z_i[r]$ for $i \in \{1,\ldots,L\}$ by:

$$Z_i[r+1] = \max[Z_i[r] + energy_i[r] - P_{av,i}T[r], 0] \quad (40)$$

Every frame r, observe $\boldsymbol{\eta}[r]$ and the queues $Z_i[r]$, and choose the device $i \in \mathcal{H}[r]$ with the smallest value of the following quantity (breaking ties arbitrarily):

$$VT_i[r] + Z_i[r][e_{i,transmit}[r] - P_{av,i}T_i[r]] - \sum_{j\neq i} Z_j[r]P_{av,j}T_i[r]$$

where $T_i[r]$ is defined as the frame size given that device i is chosen on frame r:

$$T_i[r] \triangleq T_{query} + T_{i,transmit}[r]$$

Then update the virtual queues by (40).

Note that this algorithm requires no knowledge of the probability distributions associated with files, file subsets, transmit powers, and transmit times.

B. Group Shuttle Service in a Transportation System

Here we consider an example of scheduling for a transportation system. Related scheduling problems occur at data centers (see [32][33] for competitive ratio approaches), and a rolling horizon approach to taxi dispatching is treated in [34]. Here we present a simple renewal formulation (see also [35]).

Consider a taxi driver that repeatedly takes customers from the airport to their desired local destinations. The taxi driver is equipped with a smartphone that can run shortest path algorithms over the city streets. Specifically, given an ordered pair of locations (a, b), the smartphone outputs a delay D_{ab} of a shortest path from a to b. While this delay D_{ab} typically represents an *average*, to simplify notation in this example we assume this number D_{ab} is achieved deterministically if we take the corresponding shortest path. Assume the taxi can fit at most 2 customers per ride. Renewals are defined when the taxi returns to the airport. Assume there are always at least two customers available, and these customers are new upon every renewal (so that any customer not taken on the previous frame does not wait for the next frame, but leaves by some other means). Further assume that each customer pays the same fee of P dollars, regardless of the location. The initial information is a vector $\boldsymbol{\eta}[r] = (dest_1[r], dest_2[r])$, specifying the particular destinations for the first two customers waiting in line at renewal time r. Customer 1 gets preference and is always served, whereas customer 2 may not be served. Specifically, there are three possible actions $\alpha[r]$:

- Action 1: Drive customer 1 to destination 1 using a shortest path to and from. Let $W_{01}[r]$ be the delay from airport to destination, and $W_{10}[r]$ the delay from destination to airport. This action creates an earning of P dollars, creates a delay $W_{01}[r]$ for the customer, and has frame size $T[r] = W_{01}[r] + W_{10}[r]$.
- Action 2: Drive customers 1 and 2, dropping customer 1 off first. Let $W_{01}[r]$ be the delay from airport to destination 1, $W_{12}[r]$ be the delay from destination 1 to destination 2, and $W_{20}[r]$ be the delay from destination 2 to airport. This action creates an earning of $2P$ dollars, creates a sum delay of $W_{01}[r] + (W_{01}[r] + W_{12}[r])$ for the customers, and has frame size $T[r] = W_{01}[r] + W_{12}[r] + W_{20}[r]$.
- Action 3: Drive customers 1 and 2, but first drop off customer 2. This creates an earning of $2P$, sum customer delay of $W_{02}[r] + (W_{02}[r] + W_{21}[r])$, and frame size $T[r] = W_{02}[r] + W_{21}[r] + W_{10}[r]$.

For each customer c that is actually served, define W_c^{min} as the *minimum possible delay* this customer can experience. If the taxi takes 1 customer on a given frame, then this customer is served at its minimum delay. If the taxi takes 2 customers, one is served at its minimum delay and the other is not. The goal is to maximize time average profit subject to an average delay of served customers that is at most $1+\beta$ times the average of the minimum possible delays over all served customers, where β is a given positive constant.

To this end, define the following attributes:

$$\begin{aligned} y_0[r] &\triangleq (-1) \times \text{ total earning on frame } r \\ delay_{sum}[r] &\triangleq \text{sum delay of customers on frame } r \\ T[r] &\triangleq \text{frame size for frame } r \\ N[r] &\triangleq \text{number of customers served on frame } r \\ min_{sum}[r] &\triangleq \text{sum of min possible delays of customers served on frame } r \end{aligned}$$

These attributes are deterministic functions of $\boldsymbol{\eta}[r]$ and the action taken. The average customer delay is the limit of the total sum delay of all customers served divided by the total number of customers served:

$$\lim_{R\to\infty} \frac{\sum_{r=0}^{R-1} delay_{sum}[r]}{\sum_{r=0}^{R-1} N[r]} = \frac{\overline{delay}_{sum}}{\overline{N}}$$

where $\overline{delay}_{sum}$ and $\overline{N}$ represent frame averages. Likewise, the average of the minimum possible delays of all served customers is:

$$\lim_{R\to\infty} \frac{\sum_{r=0}^{R-1} min_{sum}[r]}{\sum_{r=0}^{R-1} N[r]} = \frac{\overline{min}_{sum}}{\overline{N}}$$

The problem is then mathematically expressed by:

$$\begin{aligned} &\text{Minimize:} && \overline{y}_0/\overline{T} \\ &\text{Subject to:} && \overline{delay}_{sum}/\overline{N} \leq (1+\beta)\overline{min}_{sum}/\overline{N} \end{aligned}$$

This can be transformed into the standard form by:

$$\begin{aligned} &\text{Minimize:} && \overline{y}_0/\overline{T} \\ &\text{Subject to:} && (\overline{delay}_{sum} - (1+\beta)\overline{min}_{sum})/\overline{T} \leq 0 \end{aligned}$$

The alternative algorithm with time averaging, in Section VI-B, becomes: Define a virtual queue $Z[r]$ with dynamics:

$$Z[r+1] = \max[Z[r] + delay_{sum}[r] - (1+\beta)min_{sum}[r], 0] \quad (41)$$

Define $\theta[0] = 0$, and for $r \geq 0$ define $\theta[r+1]$ by:

$$\theta[r+1] = \sum_{k=0}^{r} y_0[k] / \sum_{k=0}^{r} T[k] \quad (42)$$

Every frame $r \in \{0, 1, 2, \ldots\}$, observe $Z[r]$, $\theta[r]$, and $\boldsymbol{\eta}[r]$ (specifying the two new requested destinations). Then run the shortest path algorithm to find delays $W_{01}[r]$, $W_{10}[r]$, $W_{12}[r]$, $W_{21}[r]$, $W_{02}[r]$, $W_{20}[r]$. Each of the three possible actions yields determined values for $y_0[r]$, $delay_{sum}[r]$, $T[r]$, $min_{sum}[r]$ as defined above. Choose the action that minimizes the following quantity (breaking ties arbitrarily):

$$V(y_0[r] - \theta[r]T[r]) + Z[r](delay_{sum}[r] - (1+\beta)min_{sum}[r])$$

At the end of each frame r, update the virtual queue $Z[r]$ by (41), and $\theta[r]$ by (42), to obtain values for $r+1$.

Note that this algorithm does not require knowledge of the probability distributions for customer locations, or of the ways in which these distributions affect the joint delay distributions for $(W_{ij}[r])$ through the street map of the city.

C. Quality-Aware Task Processing Networks

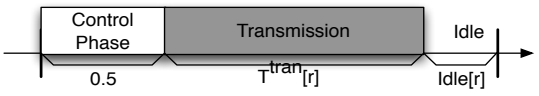


Fig. 2. An illustration of the 3 phases of a renewal frame $r \in \{0, 1, 2, \ldots\}$.

Consider a system that processes tasks with the help of 5 wireless devices. Each new task represents an event that is sensed by the devices, each at different sensing qualities. The controller must select which device reports the event information. This is similar to the reporting model of [19], with the exception that here we consider the reporting time. The renewal structure is shown in Fig. 2. At the beginning of each new task r, a period of 0.5 time units is expended to communicate control information. Each of the 5 devices expends 0.5 units of energy in this control phase. At the end of this phase, the controller obtains a vector $\boldsymbol{\eta}[r]$ of parameters for task r. The vector $\boldsymbol{\eta}[r]$ has the form:

$$\boldsymbol{\eta}[r] = [(qual_1[r], T_1^{tran}[r]), \ldots, (qual_5[r], T_5^{tran}[r])]$$

where for each $l \in \{1, \ldots, 5\}$, $qual_l[r]$ is a real number representing the *information quality* if device l is chosen to process task r, and $T_l^{tran}[r]$ is the *transmission time* required for device l to transmit the corresponding information to a receiving station. The controller chooses one of the 5 devices to process the task. It also chooses the amount of *idle time* at the end of the frame, chosen within the interval $[0, I^{max}]$ for some constant $I^{max} > 0$ (see Fig. 2). Thus, the action $\alpha[r] \in \mathcal{A}_{\boldsymbol{\eta}[r]}$ has the form:

$$\alpha[r] = (l[r], Idle[r]) \in \{1, 2, 3, 4, 5\} \times \{I \in \mathbb{R} | 0 \leq I \leq I^{max}\}$$

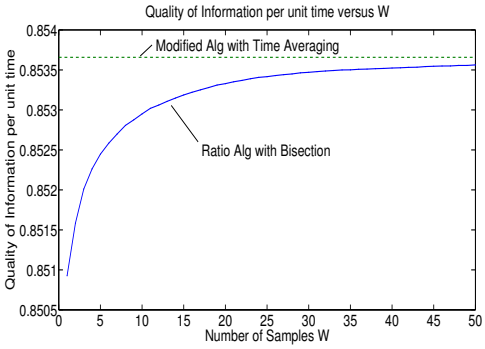


Fig. 3. Utility for the drift-plus-penalty ratio algorithm (with bisection) and the alternative algorithm that uses time averaging.

Define P^{tran} as the power expenditure associated with wireless transmission. The chosen device $l[r]$ expends $P^{tran} \times T_{l[r]}^{tran}$ units of energy in the transmit phase, while all other devices $l \neq l[r]$ expend no energy in this phase. None of the devices expend energy in the idle phase, which helps to limit the average power expenditure in the system.

The goal is to maximize *quality of information (q.o.i) per unit time* subject to an average power constraint of 0.25 at each device. Define $\tilde{y}_0(\boldsymbol{\eta}[r], \alpha[r])$ as -1 times the q.o.i. obtained for task r, $\tilde{y}_l(\boldsymbol{\eta}[r], \alpha[r])$ as the energy expended by device l on task r, and $\tilde{T}(\boldsymbol{\eta}[r], \alpha[r])$ as the frame size for task r:

$$\begin{aligned}
\tilde{y}_0(\boldsymbol{\eta}[r], \alpha[r]) &\triangleq -\sum_{l=1}^{5} qual_l[r] 1_{\{l[r]=l\}} \\
\tilde{y}_l(\boldsymbol{\eta}[r], \alpha[r]) &\triangleq 0.5 + P^{tran} T_l^{tran}[r] 1_{\{l[r]=l\}} \\
&\quad \forall l \in \{1, \ldots, 5\} \\
\tilde{T}(\boldsymbol{\eta}[r], \alpha[r]) &\triangleq 0.5 + \sum_{l=1}^{5} T_l^{tran}[r] 1_{\{l[r]=l\}} + Idle[r]
\end{aligned}$$

where $1_{\{l[r]=l\}}$ is an indicator function that is 1 if $l[r] = l$ and 0 else. The problem is then to minimize $\overline{y}_0/\overline{T}$ subject to $\overline{y}_l/\overline{T} \leq 0.25$ for all $l \in \{1, \ldots, 5\}$.

We simulate the drift-plus-penalty ratio algorithm for 10^6 frames, using the bisection method with W past samples of $\boldsymbol{\eta}[r]$ as in (34) of Section V-C. We use $P^{tran} = 1.0$, $I^{max} = 5.0$. The vectors $\{\boldsymbol{\eta}[r]\}_{r=0}^{\infty}$ are assumed to be i.i.d. with independently chosen components, where $T_l^{tran}[r]$ is uniformly distributed in $[0.5, 2.5]$ for all l, and $qual_l[r]$ is uniformly distributed in $[0, l]$ for $l \in \{1, 2, 3, 4, 5\}$ (so that device 5 tends to have the highest quality, while device 1 tends to have the lowest). Each step of the bisection computes $val(\theta)$ in (34) according to a simple deterministic optimization. In particular, for the wth term in $val(\theta)$, corresponding to sample $\boldsymbol{\eta}_w$, choose action α to minimize:

$$V\tilde{y}_0(\alpha, \boldsymbol{\eta}_w) + \sum_{l=1}^{5} Z_l[r]\tilde{y}_l(\alpha, \boldsymbol{\eta}_w) - \theta\tilde{T}(\alpha, \boldsymbol{\eta}_w)$$

This amounts to choosing $Idle[r, w] = 0$ whenever $\theta \leq 0$, and $Idle[r, w] = I^{max}$ else. Further, it chooses $l[r, w]$ as the index $l \in \{1, \ldots, 5\}$ that minimizes:

$$-V qual_l[r, w] + (Z_l[r]P^{tran} - \theta)T_l^{tran}[r, w]$$

At the beginning of each frame r we initialize $\theta_{min} = -5V$, $\theta_{max} \triangleq \sum_{l=1}^{5} Z_l[r]3$, which can be shown to always satisfy $val(\theta_{min}) \geq 0$ and $val(\theta_{max}) \leq 0$. The bisection routine is run until $\theta_{max} - \theta_{min} < 0.001$. Using $V = 100$, the resulting q.o.i per unit time is plotted in Fig. 3. This increases to its

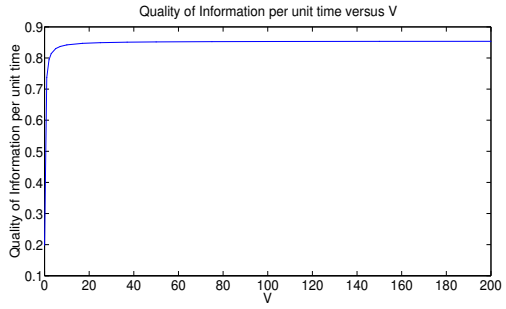
optimal value as W is increased. However, in this example, W does not need to be very large for accurate results: Even $W = 1$ produces a value that is near optimal (note that the y-axis in Fig. 3 distinguishes only in the 3rd significant digit). All average power constraints are met in all simulations (for each W). Results for $W = 10$ are: $q.o.i/\overline{T} = 0.852950$, $\overline{T} = 3.180275$, $\overline{Idle} = 1.421260$, $\overline{y}_0 = -2.712615$, and:

$$\overline{y}_1/\overline{T} = 0.182335$$
$$\overline{y}_2/\overline{T} = 0.249547 \ , \ \overline{y}_3/\overline{T} = 0.250018$$
$$\overline{y}_4/\overline{T} = 0.2500320.25 \ , \ \overline{y}_5/\overline{T} = 0.250046$$

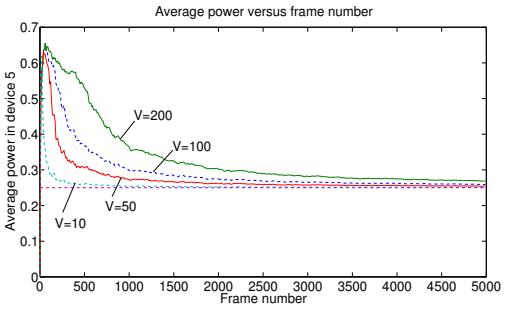
Notice that devices $\{2, 3, 4, 5\}$ are utilized to their maximum power constraint 0.25 because these tend to give the highest quality, while average power for device 1 is slack.

The alternative algorithm of Section VI-B, which does not require a bisection routine and amounts to a simple deterministic optimization for (39) every frame, achieves similar time average power expenditures to the above. It also achieves utility as shown in Fig. 3, being the constant that does not depend on W (as no sampling from the past is needed). Its utility is slightly larger than that of the bisection algorithm, and is approached by the bisection algorithm as W increases. It appears that this algorithm is simpler and yields "automatic learning" by using the time average value $\theta[r]$, but it might have trouble adapting if system parameters change.

To illustrate the performance tradeoff with V, Fig. 4(a) shows average quality of information for the ratio algorithm with bisection, using $W = 10$, but varying V from 0 to 200. Quality quickly improves with V. In this case, it can be shown that a Slater condition holds (so that all power constraints can be achieved with slackness), so that average virtual queue size is $O(V)$, and constraint violation decays like V/R rather than $\sqrt{V/R}$. The constraint violation for device 5 is illustrated versus the number of frames in Fig. 4(b). Average power converges to its constraint 0.25 as time progresses, with faster convergence for smaller values of V.



(a)



(b)

Fig. 4. (a) Time average quality of information versus V for the ratio algorithm with bisection ($W = 10$). (b) Sample path of average power for device 5 in the same experiment, showing convergence towards 0.25.

D. Comparison to the Linear Fractional Program Method

Consider an attempt to use the linear fractional program of Section III to solve the quality-aware scheduling example of the previous subsection. This approach requires $\boldsymbol{\eta}[r]$ to take values in a finite set, and requires full knowledge of the corresponding probability mass function $q(\boldsymbol{\eta})$. However, in this example, $\boldsymbol{\eta}[r]$ is a 10-dimensional vector of real numbers. Hence, it has an infinite state space. One might approximate the problem by quantizing each component of $\boldsymbol{\eta}[r]$ to one of 100 possibilities. This yields a state space of size 10^{100}, and the resulting linear fractional program would have so many variables that it could not be practically solved.

Instead, here we consider a modified system where the controller makes decisions on each frame without knowledge of $\boldsymbol{\eta}[r]$. This precludes opportunistic scheduling and fundamentally degrades system performance. However, the resulting linear fractional program can be solved in this case. Numerically solving shows that the optimal randomized algorithm uses $I[r] = 1.66655$ for all frames r, and independently chooses $\alpha[r] \in \{1, 2, 3, 4, 5\}$ every frame with probabilities $Pr[\alpha = 1] = 0$, $Pr[\alpha = 2] = 0.16666$, $Pr[\alpha = 3] = Pr[\alpha = 4] = Pr[\alpha = 5] = 0.27778$. The resulting optimal quality of information per unit time is $\overline{y}_0/\overline{T} = 0.5$, which is significantly lower than the value 0.852950 achievable by opportunistic scheduling (see also Fig. 5).

We compare the non-opportunistic randomized algorithm that uses these pre-computed probabilities to the non-opportunistic online dynamic approach. The dynamic approach uses knowledge of the expected penalty and frame sizes for each decision option. It chooses $l[r]$, $Idle[r]$ every frame r to minimize $[-V\hat{q}(l[r]) + \sum_{i=1}^{5} Z_i[r]\hat{y}_i(l[r])]/[2 + Idle[r]]$, where $\hat{q}(l[r]) = l/2$ for $l \in \{1, 2, 3, 4, 5\}$, and $\hat{y}_i(l[r])$ is the average power expended by device i if device $l[r]$ is chosen for transmission. Fig. 5 shows that the $\overline{y}_0/\overline{T}$ of the dynamic algorithm quickly approaches the optimal value of 0.5 as V is increased. Fig. 6 shows the constraint violations at device 4 for different values of V. The dynamic algorithm provides tighter constraint satisfaction for $V \leq 50$ because it bases decisions on the virtual queues (and hence on the current constraint violations), whereas the randomized algorithm makes memoryless decisions that lead to a larger variance.

VIII. Conclusion

This paper develops a method for optimizing time averages in general renewal systems. Every renewal frame, a policy is chosen that affects the frame size and a penalty vector. A dynamic algorithm is developed for minimizing the time average of one penalty subject to time average constraints on the others. This work extends the theory of Lyapunov optimization to treat systems with variable frame lengths. It can be applied to a variety of systems, including peer-to-peer

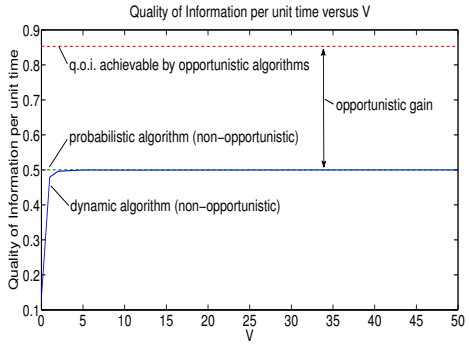


Fig. 5. Quality of information versus V for the non-opportunistic scheduling setting. The gain achievable by opportunistic scheduling is also illustrated.

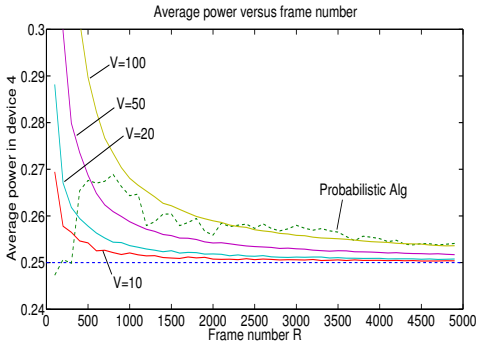


Fig. 6. Average power in device 4 as a function of time for the probabilistic algorithm and the dynamic algorithms with $V \in \{10, 20, 50, 100\}$.

networks, task processing networks, transportation systems, and large classes of Markov decision problems.

REFERENCES

[1] M. J. Neely. Dynamic optimization and learning for renewal systems. *Proc. Asilomar Conf. on Signals, Systems, and Computers*, pp. 681-690, Nov. 2010.

[2] R. Gallager. *Discrete Stochastic Processes*. Kluwer Academic Publishers, Boston, 1996.

[3] S. Ross. *Introduction to Probability Models*. Academic Press, 8th edition, Dec. 2002.

[4] L. Georgiadis, M. J. Neely, and L. Tassiulas. Resource allocation and cross-layer control in wireless networks. *Foundations and Trends in Networking*, vol. 1, no. 1, pp. 1-149, 2006.

[5] M. J. Neely. Energy optimal control for time varying wireless networks. *IEEE Transactions on Information Theory*, vol. 52, no. 7, pp. 2915-2934, July 2006.

[6] W. B. Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. John Wiley & Sons, 2007.

[7] D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, Belmont, Mass, 1996.

[8] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[9] E. Altman. *Constrained Markov Decision Processes*. Boca Raton, FL, Chapman and Hall/CRC Press, 1999.

[10] S. Meyn. *Control Techniques for Complex Networks*. Cambridge University Press, 2008.

[11] V. S. Borkar. An actor-critic algorithm for constrained Markov decision processes. *Systems and Control Letters (Elsevier)*, vol. 54, pp. 207-213, 2005.

[12] N. Salodkar, A. Bhorkar, A. Karandikar, and V. S. Borkar. An on-line learning algorithm for energy efficient delay constrained scheduling over a fading channel. *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 4, pp. 732-742, May 2008.

[13] D. V. Djonin and V. Krishnamurthy. q-learning algorithms for constrained markov decision processes with randomized monotone policies: Application to mimo transmission control. *IEEE Transactions on Signal Processing*, vol. 55, no. 5, pp. 2170-2181, May 2007.

[14] F. J. Vázquez Abad and V. Krishnamurthy. Policy gradient stochastic approximation algorithms for adaptive control of constrained time varying markov decision processes. *Proc. IEEE Conf. on Decision and Control*, pp. 2823-2828, Dec. 2003.

[15] F. Fu and M. van der Schaar. A systematic framework for dynamically optimizing multi-user video transmission. *IEEE Journal on Selected Areas in Communications*, vol. 28, no. 3, pp. 308-320, April 2010.

[16] F. Fu and M. van der Schaar. Decomposition principles and online learning in cross-layer optimization for delay-sensitive applications. *IEEE Trans. Signal Processing*, vol. 58, no. 3, pp. 1401-1415, March 2010.

[17] M. J. Neely. Stochastic optimization for markov modulated networks with application to delay constrained wireless scheduling. *Proc. IEEE Conf. on Decision and Control (CDC)*, Shanghai, China, pp. 4826-4833, Dec. 2009.

[18] C. Li and M. J. Neely. Network utility maximization over partially observable Markovian channels. *Proc. Intl. Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, pp. 17-24, May 2011.

[19] B. Liu, P. Terlecky, A. Bar-Noy, R. Govindan, M. J. Neely, and D. Rawitz. Optimizing information credibility in social swarming applications. *IEEE Trans. on Parallel and Distributed Systems*, vol. 23, no. 6, pp. 1147-1158, June 2012.

[20] L. Jiang and J. Walrand. *Scheduling and Congestion Control for Wireless and Processing Networks*. Morgan & Claypool, 2010.

[21] L. Huang and M. J. Neely. Utility optimal scheduling in processing networks. *Performance Evaluation*, vol. 68, no. 11, pp. 1002-1021, Nov. 2011.

[22] L. Tassiulas and A. Ephremides. Dynamic server allocation to parallel queues with randomly varying connectivity. *IEEE Transactions on Information Theory*, vol. 39, no. 2, pp. 466-478, March 1993.

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

[24] M. J. Neely. Dynamic optimization and learning for renewal systems. *Arxiv Technical Report, arXiv:1011.5942v1*, Nov. 2010.

[25] M. J. Neely. Stability and probability 1 convergence for queueing networks via Lyapunov optimization. *Journal of Applied Mathematics*, vol. 2012, doi:10.1155/2012/831909, 2012.

[26] M. J. Neely, S. T. Rager, and T. F. La Porta. Max weight learning algorithms for scheduling in unknown environments. *IEEE Transactions on Automatic Control*, to appear.

[27] C.-P. Li and M. J. Neely. Delay and rate-optimal control in a multi-class priority queue with adjustable service rates. *Proc. IEEE INFOCOM*, pp. 2976-2980, 2012.

[28] R. Urgaonkar and M. J. Neely. Opportunistic cooperation in cognitive femtocell networks. *IEEE Journal on Selected Areas in Communications*, vol. 30, no. 3, pp. 607-616, April 2012.

[29] D. P. Bertsekas. *Dynamic Programming and Optimal Control, vols. 1 and 2*. Athena Scientific, Belmont, Mass, 1995.

[30] K. Doppler, M. Rinne, C. Wijting, C. B. Ribeiro, and K. Hugl. Device-to-device communication as an underlay to LTE-advanced networks. *IEEE Comm Mag*, pp. 42-49, Dec. 2009.

[31] M. J. Neely. Wireless peer-to-peer scheduling in mobile networks. *Proc. 46th Conf. on Information Sciences and Systems (CISS)*, March 2012.

[32] M. Lin, A. Wierman, L. L. H. Andrew, and E. Thereska. Dynamic right-sizing for power-proportional data centers. *Proc. IEEE INFOCOM*, pp. 1098-1106, 2011.

[33] T. Lu and M. Chen. Simple and effective dynamic provisioning for power-proportional data centers. *Proc. 46th Conf. on Information Sciences and Systems (CISS)*, March 2012.

[34] K. I. Wong and M. G. H. Bell. The optimal dispatching of taxis under congestion: A rolling horizon approach. *Journal of Advanced Transportation*, vol. 40, no. 2, pp. 203-220, Spring/Summer 2006.

[35] M. J. Neely. Asynchronous scheduling for energy optimality in systems with multiple servers. *Proc. 46th Conf. on Information Sciences and Systems (CISS)*, March 2012.