Online Fractional Programming for Markov Decision Systems

Michael J. Neely

***Abstract*— We consider a system with K states which operates over frames with different lengths. Every frame, the controller observes a new random event and then chooses a control action based on this observation. The current state, random event, and control action together affect: (i) the frame size, (ii) a vector of penalties incurred over the frame, and (iii) the transition probabilities to the next state visited at the end of the frame. The goal is to minimize the time average of one penalty subject to time average constraints on the others. This problem has applications to task scheduling in computer systems and wireless networks, where each task can take a different amount of time and may change the state of the network. An example is energy-optimal scheduling in a system with several energy-saving transmission modes, where transitions to a different mode incur energy and/or delay penalties. We pose the problem as a stochastic linear fractional program and present an online Lyapunov drift method for solving it. For large classes of problems, the solution can be implemented without any knowledge of the random event probabilities.**

***Index Terms*— Dynamic scheduling, stochastic control, energy savings, computer networks, wireless networks**

I. Introduction

We consider dynamic scheduling in a system with variable length frames. The system is in one of K states at the beginning of each frame, and operates according to a generalization of a Markov decision process. Specifically, at the beginning of each frame $r \in \{0, 1, 2, \ldots\}$, the controller observes the current system state $k[r]$ and a random system event $\omega[r]$, and makes a control action $\alpha[r]$ based on these observations. The action $\alpha[r]$ is selected from an abstract set $\mathcal{A}(k[r], \omega[r])$ that possibly depends on $k[r]$ and $\omega[r]$. The $\alpha[r]$ decision affects:

1) The size $T[r]$ of frame r. We have $T[r] = \hat{T}(k[r], \omega[r], \alpha[r])$, where $\hat{T}(k, \omega, \alpha)$ is a function of state k, action α, and event ω.
2) The transition probabilities $P_{ij}[r]$ for frame r, where $i = k[r]$ is the current state, held for the duration of frame r, and j is a possible next state visited at the end of the frame. We have $P_{ij}[r] = \hat{P}_{ij}(\omega[r], \alpha[r])$, where $\hat{P}_{ij}(\omega, \alpha)$ is a function of states i, j, event ω, and action α.
3) A vector $\boldsymbol{y}[r] = (y_0[r], y_1[r], \ldots, y_L[r])$ of penalties for frame r. For each $l \in \{0, 1, \ldots, L\}$ we have $y_l[r] = \hat{y}_l(k[r], \omega[r], \alpha[r])$, where $\hat{y}_l(k, \omega, \alpha)$ is a function of state k, event ω, and action α.

The author is with the Electrical Engineering department at 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.

We assume throughout that $\hat{T}(k, \omega, \alpha)$, $\hat{P}_{ij}(\omega, \alpha)$, and $\hat{y}_l(k, \omega, \alpha)$ are known deterministic functions, although in some cases our algorithm does not require full knowledge of these functions, and can extend to random functions. We assume throughout that $\omega[r]$ takes values in an abstract set Ω of arbitrary cardinality, and is i.i.d. over frames with an unknown probability distribution $p(\omega)$. For example, the $\omega[r]$ process can represent a vector of random arrivals or channel states for a wireless system, and can possibly be null if such a process is not relevant to the system of interest. One could also remove $\omega[r]$ by incorporating it into the current state $k[r]$, but this would create a much larger (possibly infinite) state space. Our solution is considerably simplified when $\omega[r]$ is treated separately. Indeed, the complexity of our solution does not depend on the size of the set Ω.

The goal is to minimize the time average associated with penalty $y_0[r]$ subject to time average constraints on $y_l[r]$ for $l \in \{1, \ldots, L\}$. Specifically, for each integer $R > 0$ define $\overline{y}_l[R]$ and $\overline{T}[R]$ by:

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

where the expectations are with respect to the random events that occur and the (possibly randomized) control algorithm that is used. For simplicity of exposition, assume the control algorithm leads to time average expectations that converge to well defined limits $\overline{y}_l$ and $\overline{T}$ as $R \to \infty$. We seek to solve the problem:[1]

$$\text{Minimize:} \quad \overline{y}_0/\overline{T} \tag{1}$$

$$\text{Subject to:} \quad \overline{y}_l/\overline{T} \leq c_l \ \ \forall l \in \{1, \ldots, L\} \tag{2}$$

$$\alpha[r] \in \mathcal{A}(k[r], \omega[r]) \ \ \forall r \in \{0, 1, 2, \ldots\} \tag{3}$$

where c_l are given constants for $l \in \{1, \ldots, L\}$. Similar to basic renewal theory, the quantity $\overline{y}_l/\overline{T}$ represents the time average associated with the $y_l[r]$ process [1]. The above can also be used to solve modified problems of minimizing $\overline{y}_0$ subject to $\overline{y}_l \leq c_l$. This is done simply by defining a "virtual" frame size $T[r]$ that is equal to 1 for all frames r. This model can be used for a variety of systems, with examples given in the next two subsections.

A. Discrete Time Controlled Markov Chain Example

Suppose we have a controlled Markov chain with K states that evolves in discrete time over slots $r \in \{0, 1, 2, \ldots\}$. All

[1]The assumption that the limits exist is used to simplify discussion. Our theorems use lim sups where appropriate.

frames have size $T[r] = 1$, called a *time slot*. Every slot r, given that we are in state $k[r]$, we observe a random event $\omega[r]$ and make a control decision $\alpha[r] \in \mathcal{A}(k[r], \omega[r])$ that affects the transition probabilities to the next state and also affects a vector of penalties $\boldsymbol{y}[r]$, which are deterministic functions of $k[r]$, $\omega[r]$, $\alpha[r]$. In the absence of observed events $\omega[r]$ and when the action space $\mathcal{A}(k, \omega)$ is a finite set for all k, ω, the resulting Markov decision problem (MDP) can be solved by the conventional linear programming method [2][3]. The technique we present in this paper provides a method that allows infinite action spaces, and possibly an infinite event space for $\omega[r]$, and does not require knowledge of the probability distribution associated with $\omega[r]$.

B. Energy-Efficient Scheduling Example

Suppose we have a computer system that has K processing modes, each with different energy and processing rate properties. For example, there can be an "idle" mode which uses minimum power and does not allow any processing. There are L job classes. Every frame r we choose which single class $l[r] \in \{1, \ldots, L\}$ to serve, spending one unit of time in processing mode $k[r]$ (given that $k[r]$ is the processing mode at the beginning of the frame). We also decide whether to stay in state $k[r]$ on frame $r+1$, or to transition to some new state, which incurs extra time. Let $T_{kj}(l)$ denote the total frame size given we are initially in state k, choose to serve class l, and transition to state j. The total energy expended by this choice is $e_{kj}(l)$. The constants $T_{kj}(l)$ and $e_{kj}(l)$ are assumed known. Further, new jobs with fixed workload arrive according to a Poisson process of rates $\lambda_1, \ldots, \lambda_L$, so that $T_{kj}(l)\lambda_m$ is the average number of new jobs of type m that arrive over a frame of size $T_{kj}(l)$.

Thus, there is no random event $\omega[r]$ at the beginning of a frame r. The control decision $\alpha[r]$ has the form $\alpha[r] = (l[r], j[r])$, where $l[r] \in \{1, \ldots, L\}$ and is the class of data served on frame r, and $j[r]$ is the choice of next-state at the end of the frame. We then have:

$$\begin{aligned} T[r] &= T_{k[r]j[r]}(l[r]) \\ P_{k[r]n}[r] &= \begin{cases} 1 & \text{if } n = j[r] \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

Define $y_0[r]$ as the energy expended in frame r. We can thus formulate the problem of choosing a control action every frame to minimize average power $\overline{y}_0/\overline{T}$ subject to serving all traffic classes at a rate at least ϵ beyond the input rate λ_l, for some pre-specified $\epsilon \geq 0$. These constraints are enforced by defining for each $l \in \{1, \ldots, L\}$:

$$y_l[r] = \lambda_l T_{k[r]j[r]}(l[r]) - 1_{\{l[r]=l\}}$$

which is the average arrivals of class l over frame r, minus 1 if we serve class l on frame r. The desired constraints are thus enforced by $\overline{y}_l/\overline{T} \leq -\epsilon$. We consider this special case in more detail later and show that, while the $y_l[r]$ penalties are defined in terms of the λ_l rates, the algorithm does not require knowledge of these rates.

C. Prior Work on Dynamic Markov Decision Policies

Markov decision problems (MDPs) are typically analyzed in the context of 1-slot frames, and without the additional $\omega[r]$ event process. The simplest approach is to write the problem as an offline linear program, assuming all probabilities are known and the action space is finite [2][3]. Neuro-dynamic programming and q-learning methods for unconstrained problems are given in [4]. Extensions to constrained problems are treated in [5][6] via 2-timescale arguments, stochastic approximation, and fluid limits. Related applications to wireless systems are treated in [7][8][9][10][11]. These methods typically require finite action spaces, do not have the random event process $\omega[r]$, and do not have variable frame sizes.

Our approach is the most closely related to [5], which uses an online 2-timescale method to solve the linear program corresponding to the discrete time MDP. However, in our variable frame length context, linear programs must be replaced by linear fractional programs. Further, we use a fundamentally different technique that allows for infinite action spaces and random event processes $\omega[r]$, where the solution does not require knowledge of the $p(\omega)$ probabilities and has complexity that does not depend on the number of possible $\omega[r]$ outcomes. While our approach does not involve two timescales, it does break the problem into two separate stages of algorithms: The first algorithm provides a provably fast online learning of parameters associated with the *optimal solution*. The second algorithm uses these parameters in an online implementation. These algorithms can be run in parallel, although we analyze them separately in this paper. We use Lyapunov optimization techniques from [12][1] together with MDP concepts. Related work in [11] uses Lyapunov optimization over multi-slot intervals, using a forced renewal assumption and a stochastic shortest path solver on each interval. The current paper is distinct from [11] in that it does not require forced renewals, and it makes greedy decisions on each frame, rather than using a stochastic shortest path solver over multiple frames.

II. The Linear Fractional Program

For simplicity of exposition, first assume there are no events $\omega[r]$, so that action spaces $\mathcal{A}(k, \omega)$ can be written as $\mathcal{A}(k)$, and similarly we use notation $\hat{y}_l(k, \alpha)$, $\hat{T}(k, \alpha)$, $\hat{P}_{ij}(\alpha)$. Further assume $\mathcal{A}(k)$ is a finite set for each $k \in \{1, \ldots, K\}$. A stationary and randomized policy is characterized by a conditional probability distribution $\beta(\alpha|k)$ for $k \in \{1, \ldots, K\}$ and $\alpha \in \mathcal{A}(k)$. Such a policy observes $k[r]$ at the beginning of each frame r and then chooses an action $\alpha[r] \in \mathcal{A}(k[r])$ with probability $\beta(\alpha[r]|k[r])$. It can be shown that the problem (1)-(3) can be solved over the class of stationary and randomized algorithms, where the optimal probabilities $\beta(\alpha|k)$ are given in terms of values $\phi(k, \alpha)$ by:

$$\beta(\alpha|k) = \frac{\phi(k, \alpha)}{\sum_{x \in \mathcal{A}(k)} \phi(k, x)}$$

where $\phi(k, \alpha)$ represents the steady state probability of being in state $k[r] = k$ and choosing action $\alpha[r] = \alpha$. These values are optimally solved by the following *linear fractional*

program:

Minimize:
$$\frac{\sum_{k=1}^{K}\sum_{\alpha\in\mathcal{A}(k)}\phi(k,\alpha)\hat{y}_0(k,\alpha)}{\sum_{k=1}^{K}\sum_{\alpha\in\mathcal{A}(k)}\phi(k,\alpha)\hat{T}(k,\alpha)}$$
Subject to:
$$(i)\ \frac{\sum_{k=1}^{K}\sum_{\alpha\in\mathcal{A}(k)}\phi(k,\alpha)\hat{y}_l(k,\alpha)}{\sum_{k=1}^{K}\sum_{\alpha\in\mathcal{A}(k)}\phi(k,\alpha)\hat{T}(k,\alpha)} \leq c_l\ \forall l\in\{1,\ldots,L\}$$
$$(ii)\ \sum_{\alpha\in\mathcal{A}(k)}\phi(k,\alpha) = \sum_{i=1}^{K}\sum_{\alpha\in\mathcal{A}(k)}\phi(i,\alpha)\hat{P}_{ik}(\alpha)\quad \forall k\in\{1,\ldots,K\}$$
$$(iii)\ \phi(k,\alpha)\geq 0\ \forall k\in\{1,\ldots,K\},\alpha\in\mathcal{A}(k)$$
$$(iv)\ \sum_{k=1}^{K}\sum_{\alpha\in\mathcal{A}(k)}\phi(k,\alpha)=1$$

The ratios in the above program can be viewed as representing the ratios of averages $\overline{y}_l/\overline{T}$. Constraint (ii) in the above program can be viewed as the classic *global balance equation* [2], where the left-hand-side of the equality is the steady state probability of being in state k at the beginning of a frame, and the right-hand-side is the steady state probability of transitioning to state k at the end of a frame.

Such linear fractional programs can be solved offline by converting to linear programs via a non-linear change of variables [13]. However, this may involve a very large number of variables if the sets $\mathcal{A}(k)$ are large. Further, this does not allow the possibility of an $\omega[r]$ process. Our approach develops an *online* solution technique that allows for an $\omega[r]$ process, does not require knowledge of probabilities $p(\omega)$, and allows for possibly infinite sets Ω and $\mathcal{A}(k,\omega)$.

Section III develops a solution to a generalized version of the above problem. We note that such solutions are the most meaningful when the resulting stationary policy has a single recurrent class of states, resulting in time average expectations that are the same, with probability 1, as the pure time averages. Examples of "degenerate" systems can be constructed where the above problem of optimizing time average expectations is solvable, so that time average expectations meet the desired constraints, but for which it is impossible for pure time averages to meet the constraints.

A. Optimality over Stationary Policies

Now allow an $\omega[r]$ process, and allow the sets Ω and $\mathcal{A}(k,\omega)$ to be infinite (we still assume there are only K states). The stationary and randomized policies associated with the above (finite) linear fractional program have the following generalization: Define a (k,ω)-*only policy* $\alpha^*[r]$ as one that observes $(k[r],\omega[r])$ every frame r, and independently chooses $\alpha^*[r]\in\mathcal{A}(k[r],\omega[r])$ according to a stationary probability distribution that depends only on $(k[r],\omega[r])$. Note that a given (k,ω)-only policy $\alpha^*[r]$ makes the process $k[r]$ a discrete time homogeneous Markov chain on state space $\{1,\ldots,K\}$ and with transition probability matrix $P^*=(P^*_{ij})$, where:
$$P^*_{ij} = \mathbb{E}_{\omega[r]}\left[\mathbb{E}\left[\hat{P}_{ij}(\omega[r],\alpha^*[r])|\omega[r],k[r]=i\right]\right]$$

Let $\boldsymbol{\pi}^*=(\pi^*_1,\ldots,\pi^*_K)$ be a row vector that satisfies:
$$\boldsymbol{\pi}^* = \boldsymbol{\pi}^* P^* \tag{4}$$
Such a probability vector $\boldsymbol{\pi}^*$ always exists, but may not be unique. If the underlying Markov chain has a single irreducible class, then it is unique and represents the vector of time average fractions of time being in each state. Define $ratio^{opt}$ as the infimum value of (1) under all such (k,ω)-only policies that satisfy the constraints. If the sets Ω, $\mathcal{A}(k,\omega)$ are finite, then $ratio^{opt}$ is also optimal over *all policies* (not just (k,ω)-only policies). This can be extended to infinite sets with some mild but notationally complex additional assumptions. Rather than complicate notation, we simply measure our performance with respect to $ratio^{opt}$, and make the following assumption:

Assumption 1: There exists a (k,ω)-only policy $\alpha^*[r]$, together with a probability distribution $\boldsymbol{\pi}^*$ that satisfies (4), such that:
$$\frac{\mathbb{E}\left[y^*_0[r]\right]}{\mathbb{E}\left[T^*[r]\right]} = ratio^{opt} \tag{5}$$
$$\frac{\mathbb{E}\left[y^*_l[r]\right]}{\mathbb{E}\left[T^*[r]\right]} \leq c_l\ \forall l\in\{1,\ldots,L\} \tag{6}$$
where the expectations are with respect to being in state $k[r]$ at the beginning of state r with probability distribution $\boldsymbol{\pi}^*$, and $y^*_l[r]=\hat{y}_l(k[r],\omega[r],\alpha^*[r])$, $T^*[r]=\hat{T}(k[r],\omega[r],\alpha^*[r])$.

Assumption 1 holds when problem (1)-(3) is feasible and the sets Ω, $\mathcal{A}(k,\omega)$ are finite, as well as in many other cases.

B. Boundedness Assumptions

For simplicity, we assume for each $l\in\{0,1,\ldots,L\}$ there are finite (possibly negative) constants y^{min}_l and y^{max}_l such that for all possible $k[r],\omega[r],\alpha[r]$:
$$y^{min}_l \leq \hat{y}_l(k[r],\omega[r],\alpha[r]) \leq y^{max}_l$$
Also, we assume there are constants T^{min} and T^{max} such that:
$$0 < T^{min} \leq \hat{T}(k[r],\omega[r],\alpha[r]) \leq T^{max}$$

III. Online Learning

We now write the general problem (with possibly infinite sets Ω and $\mathcal{A}(k,\omega)$) as a stochastic optimization of time averages of certain attributes. To do so, we define a modified system where the same $\omega[r]$ process is observed every frame, but with the following key difference: There are no Markovian dynamics, and the "state variable" $k[r]$ is treated as a *decision variable* that can be chosen every frame within the set $\{1,\ldots,K\}$. We shall enforce a "global balance" constraint on the average fraction of time that $k[r]=k$. After choosing $k[r]$, we also choose decision variable $\alpha[r]\in\mathcal{A}(k[r],\omega[r])$, and this affects $T[r]$, $y_l[r]$, and $P_{ij}[r]$ by the functions $\hat{T}(k[r],\omega[r],\alpha[r])$, $\hat{y}_l(k[r],\omega[r],\alpha[r])$, $\hat{P}_{ij}(\omega[r],\alpha[r])$. Define attributes $q_{ij}[r]$ for $i,j\in\{1,\ldots,K\}$ as functions of the decisions on frame r:
$$q_{ij}[r] = 1_{\{k[r]=i\}}\hat{P}_{ij}(\omega[r],\alpha[r]) \tag{7}$$
where $1_{\{k[r]=i\}}$ is an indicator function that is 1 if $k[r]=i$, and 0 else. We want to solve the following stochastic

optimization problem, where $\overline{y}_l$ and $\overline{q}_{ij}$ represent time average expectations:

$$\text{Minimize:} \quad \frac{\overline{y}_0}{\overline{T}} \tag{8}$$

$$\text{Subject to:} \quad \frac{\overline{y}_l}{\overline{T}} \leq c_l \ \forall l \in \{1,\ldots,L\} \tag{9}$$

$$\sum_{j=1}^{K} \overline{q}_{kj} = \sum_{i=1}^{K} \overline{q}_{ik} \ \forall k \in \{1,\ldots,K\} \tag{10}$$

$$k[r] \in \{1,\ldots,K\} \ \forall r \in \{0,1,2,\ldots\} \tag{11}$$

$$\alpha[r] \in \mathcal{A}(k[r],\omega[r]) \ \forall r \in \{0,1,2,\ldots\} \tag{12}$$

$$k[r] \text{ independent of } \omega[r] \ \forall r \in \{0,1,2,\ldots\} \tag{13}$$

The constraint (10) is a balance equation similar to the second constraint in the linear fractional program from Section II. The constraint (13) specifies that decision variable $k[r]$ must be chosen independently of $\omega[r]$, and is a non-standard constraint that does not arise in the time average optimization problems treated in [1][12]. It arises because our decision variable $k[r]$ for this modified problem should have similar properties to a state variable on the actual Markov system. Specifically, it ensures the conditional time averages achieved on this modified problem, given $k[r] = k$, can also be achieved on the actual Markov system. Otherwise, choosing $k[r]$ based on knowledge of $\omega[r]$ can incorrectly skew these conditional averages.

A. Virtual queues

To enforce the constraints (9), we use a virtual queue $Z_l[r]$ for each $l \in \{1,\ldots,L\}$ that is updated every frame as:

$$\begin{aligned} Z_l[r+1] &= \max\big[Z_l[r] + \hat{y}_l(k[r],\omega[r],\alpha[r]) \\ &\quad - c_l\hat{T}(k[r],\omega[r],\alpha[r]), 0\big] \end{aligned} \tag{14}$$

The intuition is that stabilizing $Z_l[r]$ implies that the average of the "arrival process" $y_l[r]$ is less than or equal to the average of the "service process" $c_l T[r]$, so that $\overline{y}_l \leq c_l\overline{T}$.

To enforce the constraints (10), we define a virtual queue $H_k[r]$ for each $k \in \{1,\ldots,K\}$, with udpates:

$$H_k[r+1] = H_k[r] + 1_{\{k[r]=k\}} - \sum_{i=1}^{K} q_{ik}[r] \tag{15}$$

where we have used the fact that $\sum_{j=1}^{K} q_{kj}[r] = 1_{\{k[r]=k\}}$, which follows from (7).

B. Drift-Plus-Penalty Ratio Method

Define $L[r]$ by:

$$L[r] = \frac{1}{2}\sum_{l=1}^{L} Z_l[r]^2 + \frac{1}{2}\sum_{k=1}^{K} H_k[r]^2$$

This is called a *Lyapunov function*. Define the *Lyapunov drift* $\Delta[r] = L[r+1] - L[r]$. Define $\boldsymbol{Q}[r] = (Z_1[r],\ldots,Z_L[r],H_1[r],\ldots,H_K[r])$ as the vector of all queue values for frame r. As in [1], our approach is to take control actions every frame r that greedily minimize a bound on the following *drift-plus-penalty ratio:*

$$\frac{\mathbb{E}\left[\Delta[r] + Vy_0[r]|\boldsymbol{Q}[r]\right]}{\mathbb{E}\left[T[r]|\boldsymbol{Q}[r]\right]}$$

where $T[r] = \hat{T}(k[r],\omega[r],\alpha[r])$, and V is a positive weight that will affect a performance tradeoff. To satisfy (13), it is essential for $k[r]$ to be chosen *first*, and then $\omega[r]$ is observed and $\alpha[r]$ is selected from the set $\mathcal{A}(k[r],\omega[r])$.

Lemma 1: For any control algorithm, we have:

$$\begin{aligned} \frac{\mathbb{E}\left[\Delta[r] + Vy_0[r]|\boldsymbol{Q}[r]\right]}{\mathbb{E}\left[T[r]|\boldsymbol{Q}[r]\right]} &\leq \frac{B}{\mathbb{E}\left[T[r]|\boldsymbol{Q}[r]\right]} \\ &+ \sum_{l=1}^{L} Z_l[r]\left[\frac{\mathbb{E}\left[\hat{y}_l(k[r],\omega[r],\alpha[r])|\boldsymbol{Q}[r]\right]}{\mathbb{E}\left[\hat{T}(k[r],\omega[r],\alpha[r])|\boldsymbol{Q}[r]\right]} - c_l\right] \\ &+ \sum_{k=1}^{K} H_k[r]\frac{\mathbb{E}\left[1_{\{k[r]=k\}} - \hat{P}_{k[r],k}(\omega[r],\alpha[r])|\boldsymbol{Q}[r]\right]}{\mathbb{E}\left[\hat{T}(k[r],\omega[r],\alpha[r])|\boldsymbol{Q}[r]\right]} \\ &+ \frac{\mathbb{E}\left[V\hat{y}_0(k[r],\omega[r],\alpha[r])|\boldsymbol{Q}[r]\right]}{\mathbb{E}\left[\hat{T}(k[r],\omega[r],\alpha[r])|\boldsymbol{Q}[r]\right]} \end{aligned} \tag{16}$$

where B is a finite constant that depends on the bounds y_l^{min}, y_l^{max}, T^{min}, T^{max}.

Proof: Omitted for brevity (see related results in [1]). □

Our algorithm, defined in the next subsection, is based on taking control actions that minimize the last three terms in the right-hand-side of (16) every frame.

C. Algorithm 1

Every frame r, observe queues $\boldsymbol{Q}[r]$. Then:

- For each $k \in \{1,\ldots,K\}$, compute $e_k[r]$, defined as the infimum of the following quantity over all policies for choosing $\alpha[r] \in \mathcal{A}(k,\omega[r])$:

$$\begin{aligned} &\frac{\mathbb{E}\left[V\hat{y}_0(k,\omega[r],\alpha[r]) + \sum_{l=1}^{L} Z_l[r]\hat{y}_l(k,\omega[r],\alpha[r])|\boldsymbol{Q}[r]\right]}{\mathbb{E}\left[\hat{T}(k,\omega[r],\alpha[r])|\boldsymbol{Q}[r]\right]} \\ &+ \frac{\mathbb{E}\left[\sum_{i=1}^{K} H_i[r](1_{\{i=k\}} - \hat{P}_{k,i}(\omega[r],\alpha[r]))|\boldsymbol{Q}[r]\right]}{\mathbb{E}\left[\hat{T}(k,\omega[r],\alpha[r])|\boldsymbol{Q}[r]\right]} \end{aligned} \tag{17}$$

- Choose $k[r]$ as the minimizer of $e_k[r]$ over all $k \in \{1,\ldots,K\}$. Then observe $\omega[r]$ and choose $\alpha[r] \in \mathcal{A}(k[r],\omega[r])$ to minimize (17), using $k = k[r]$.
- Update queues via (14) and (15).

The first two steps involve computing an infimum of a ratio of expectations to find $e_k[r]$, and then choosing the corresponding $\alpha[r]$ to minimize the ratio of expectations, given our choice of $k[r]$. Doing this exactly would require the probability distribution for $\omega[r]$. Fortunately, implementation of the above algorithm does not need to be exact. We allow our control decisions to be inexact by an additive constant $C \geq 0$. Specifically, we say the algorithm is a *C-additive approximation* to the drift-plus-penalty ratio if every frame r

we have:

$$\frac{\mathbb{E}\left[\Delta[r]+Vy_0[r]|\boldsymbol{Q}[r]\right]}{\mathbb{E}\left[T[r]|\boldsymbol{Q}[r]\right]} \leq \frac{B+C}{\mathbb{E}\left[T[r]|\boldsymbol{Q}[r]\right]} + \sum_{l=1}^{L} Z_l[r]\left[\frac{\mathbb{E}\left[\hat{y}_l(k^*[r],\omega[r],\alpha^*[r])|\boldsymbol{Q}[r]\right]}{\mathbb{E}\left[\hat{T}(k^*[r],\omega[r],\alpha^*[r])|\boldsymbol{Q}[r]\right]} - c_l\right] + \sum_{k=1}^{K} H_k[r]\frac{\mathbb{E}\left[1_{\{k^*[r]=k\}} - \hat{P}_{k^*[r],k}(\omega[r],\alpha^*[r])|\boldsymbol{Q}[r]\right]}{\mathbb{E}\left[\hat{T}(k^*[r],\omega[r],\alpha^*[r])|\boldsymbol{Q}[r]\right]} + \frac{\mathbb{E}\left[V\hat{y}_0(k^*[r],\omega[r],\alpha^*[r])|\boldsymbol{Q}[r]\right]}{\mathbb{E}\left[\hat{T}(k^*[r],\omega[r],\alpha^*[r])|\boldsymbol{Q}[r]\right]} \quad (18)$$

where $k^*[r]$ and $\alpha^*[r]$ are any other (possibly randomized) decisions that satisfy $k^*[r] \in \{1,\ldots,K\}$, $k^*[r]$ is independent of $\omega[r]$, and $\alpha^*[r] \in \mathcal{A}(k^*[r],\omega[r])$. An exact minimization yields $C = 0$. Performance of this algorithm under a C-additive approximation is analyzed in Section III-F.

C-additive approximations can be computed without knowledge of the probability distribution of $\omega[r]$ by using the *bisection method* in Chapter 7 of [1], which uses past samples. To ensure that our estimate $\tilde{e}_k[r]$ to $e_k[r]$ does not depend on $\omega[r]$, we use samples $\omega[r-1],\omega[r-2],\ldots,\omega[r-W]$, where W is the number of samples used. The *delayed queue analysis* and *max weight learning* theory in [14] shows that choosing $k[r]$ as the minimizer of the estimates $\tilde{e}_k[r]$ results in an accurate approximation for large W. Simpler and exact decisions can be made when the system has no $\omega[r]$ process, as described below.

D. Special Case without $\omega[r]$

In the special case when there is no $\omega[r]$ process (equivalently, when $\omega[r]$ is always the same value for all frames, with probability 1), it is shown in Chapter 7 of [1] that minimizing the ratio of expectations is done by deterministically choosing $k[r] \in \{1,\ldots,K\}$, $\alpha[r] \in \mathcal{A}(k[r])$ to minimize:

$$\frac{V\hat{y}_0(k[r],\alpha[r])}{\hat{T}(k[r],\alpha[r])} + \sum_{l=1}^{L} Z_l[r]\frac{\hat{y}_l(k[r],\alpha[r])}{\hat{T}(k[r],\alpha[r])} + \sum_{k=1}^{K} H_k[r]\frac{\left[1_{\{k[r]=k\}} - \hat{P}_{k[r],k}(\alpha[r])\right]}{\hat{T}(k[r],\alpha[r])} \quad (19)$$

E. Energy-Efficient Scheduling Example

Consider the energy-efficient scheduling example of Section I-B, which does not use $\omega[r]$. This example has penalties $y_0[r] = e_{k[r]j[r]}(l[r])$ and $y_l[r] = \lambda_l T_{k[r]j[r]}(l[r]) - 1_{\{l[r]=l\}}$ for $l \in \{1,\ldots,L\}$, where $j[r]$ is the choice of next-state, and $l[r]$ is the choice of which traffic class to serve. Then from (19), every frame r, we observe $\boldsymbol{Q}[r]$ and choose $k[r] \in \{1,\ldots,K\}$, $l[r] \in \{1,\ldots,L\}$, $j[r] \in \{1,\ldots,K\}$ to deterministically minimize:

$$\frac{Ve_{k[r]j[r]}(l[r])}{T_{k[r]j[r]}(l[r])} - \sum_{l=1}^{L} Z_l[r]\frac{1_{\{l[r]=l\}}}{T_{k[r]j[r]}(l[r])} + \sum_{k=1}^{K} H_k[r]\frac{\left[1_{\{k[r]=k\}} - 1_{\{j[r]=k\}}\right]}{T_{k[r]j[r]}(l[r])}$$

The above is a minimization over K^2L possible values. Note that this does not require knowledge of the job arrival rates $\lambda_1,\ldots,\lambda_L$ because these appear as coefficients multiplying the frame size in the $y_l[r]$ penalties for $l \in \{1,\ldots,L\}$.

From (15), the queue update for $H_k[r]$ for each $k \in \{1,\ldots,K\}$ becomes:

$$H_k[r+1] = H_k[r] + 1_{\{k[r]=k\}} - 1_{\{j[r]=k\}}$$

A direct implementation of the $Z_l[r]$ updates in (14) for each $l \in \{1,\ldots,L\}$ would be:

$$Z_l[r+1] = \max[Z_l[r] + \lambda_l T_{k[r]j[r]}(l[r]) - 1_{\{l[r]=l\}} + \epsilon T_{k[r]j[r]}(l[r]), 0]$$

Unfortunately, this update would require knowledge of the λ_l values. However, we can treat $\hat{y}_l(\cdot)$ as a *random function* equal to:

$$\hat{y}_l(k[r],j[r],l[r]) = \hat{A}_l(k[r],j[r],l[r]) - 1_{\{l[r]=l\}}$$

where $\hat{A}_l(k[r],j[r],l[r])$ is the random number of type l jobs that arrive in frame r, having frame size $T_{k[r]j[r]}(l[r])$. Then the average of $\hat{y}_l(\cdot)$ is equal to $\lambda_1 T_{k[r]j[r]}(l[r]) - 1_{\{l[r]=l\}}$, and the queue dynamics above can be modified to:

$$Z_l[r+1] = \max[Z_l[r] + \hat{A}_l(k[r],j[r],l[r]) - 1_{\{l[r]=l\}} + \epsilon T_{k[r]j[r]}(l[r]), 0]$$

It can be shown that this modification does not affect any of our performance results. In particular, Lemma 1 still holds, with the exception that the B constant in (16) is larger due to the variance of $\hat{A}_l(k[r],j[r],l[r])$.

F. Performance Theorem for Algorithm 1

Theorem 1: Suppose Assumption 1 holds, that all virtual queues are initially 0, and that $V \geq 0$. If $\omega[r]$ is i.i.d. over frames, and our control algorithm uses a C-additive approximation so that (18) holds every frame r (for some constant $C \geq 0$), then:

(a) For all frames $R > 0$ we have:

$$\frac{\overline{y}_0[R]}{\overline{T}[R]} \leq ratio^{opt} + \frac{B+C}{T^{min}V}$$

(b) All queues $Z_l[r]$, $H_k[r]$ are *rate stable*, meaning that:

$$\lim_{R\to\infty}\frac{Z_l[R]}{R} = 0 \ (w.p.1) \quad (20)$$

$$\lim_{R\to\infty}\frac{H_k[R]}{R} = 0 \ (w.p.1) \quad (21)$$

Further, for all frames $R > 0$:

$$\frac{\mathbb{E}\left[||\boldsymbol{Q}[R]||\right]}{R} \leq \frac{\sqrt{2[B+C+V(ratio^{opt}T^{max} - y_0^{min})]}}{\sqrt{R}} \quad (22)$$

where $||\boldsymbol{Q}[R]||$ is the Euclidean norm, so $||\boldsymbol{Q}[R]||^2 = 2L[R]$.

(c) We have:

$$\limsup_{R\to\infty}\overline{y}_l[R]/\overline{T}[R] \leq c_l \ \forall l \in \{1,\ldots,L\} \quad (23)$$

$$\lim_{R\to\infty}\sum_{j=1}^{K}[\overline{q}_{kj}[R] - \overline{q}_{jk}[R]] = 0 \ \forall k \in \{1,\ldots,K\} \quad (24)$$

and (23), (24) hold with probability 1 when expected time averages $\overline{y}_l[R]$, $\overline{T}[R]$, $\overline{q}_{ij}[R]$ are replaced with pure time averages.

The above theorem shows that we can choose the V parameter to be arbitrarily large to make the desired ratio arbitrarily close (within $O(1/V)$) to $ratio^{opt}$. The tradeoff with large V is the amount of convergence time needed, as indicated by (22) above and (29) of the proof.

Proof: (Theorem 1 part (a)) Consider the (k,ω)-only policy with corresponding probability vector $\boldsymbol{\pi}^* = (\pi_1^*, \ldots, \pi_K^*)$ from Assumption 1. Independently choose $k^*[r] \in \{1, \ldots, K\}$ according to distribution $\boldsymbol{\pi}^*$ (note that $k^*[r]$ then is indeed independent of $\omega[r]$). Choose $\alpha^*[r] \in \mathcal{A}(k^*[r], \omega[r])$ according to the decision rule $\alpha^*[r]$ from Assumption 1. Note that these decisions are made independently of queues $\boldsymbol{Q}[r]$. From (5)-(6) we thus have for all $l \in \{1, \ldots, L\}$:

$$\frac{\mathbb{E}\left[\hat{y}_0(k^*[r], \omega[r], \alpha^*[r])|\boldsymbol{Q}[r]\right]}{\mathbb{E}\left[\hat{T}(k^*[r], \omega[r], \alpha^*[r])|\boldsymbol{Q}[r]\right]} = \frac{\mathbb{E}\left[y_0^*[r]\right]}{\mathbb{E}\left[T^*[r]\right]} = ratio^{opt}$$

$$\frac{\mathbb{E}\left[\hat{y}_l(k^*[r], \omega[r], \alpha^*[r])|\boldsymbol{Q}[r]\right]}{\mathbb{E}\left[\hat{T}(k^*[r], \omega[r], \alpha^*[r])|\boldsymbol{Q}[r]\right]} = \frac{\mathbb{E}\left[y_l^*[r]\right]}{\mathbb{E}\left[T^*[r]\right]} \leq c_l$$

Recall that $\boldsymbol{\pi}^* = \boldsymbol{\pi}^* P^*$. Thus, for all $k \in \{1, \ldots, K\}$ we have:

$$\mathbb{E}\left[1_{\{k^*[r]=k\}} - \hat{P}_{k^*[r],k}(\omega[r], \alpha^*[r])|\boldsymbol{Q}[r]\right] = \pi_k^* - \sum_{i=1}^{K} \pi_i^* P_{ik}^* = 0$$

Using these identities in (18) gives:

$$\frac{\mathbb{E}\left[\Delta[r] + Vy_0[r]|\boldsymbol{Q}[r]\right]}{\mathbb{E}\left[T[r]|\boldsymbol{Q}[r]\right]} \leq \frac{B+C}{\mathbb{E}\left[T[r]|\boldsymbol{Q}[r]\right]} + Vratio^{opt}$$

Thus:

$$\mathbb{E}\left[\Delta[r] + Vy_0[r]|\boldsymbol{Q}[r]\right] \leq B + C + Vratio^{opt}\mathbb{E}\left[T[r]|\boldsymbol{Q}[r]\right]$$

Taking expectations of the above and using iterated expectations gives:

$$\mathbb{E}\left[\Delta[r]\right] + V\mathbb{E}\left[y_0[r]\right] \leq B + C + Vratio^{opt}\mathbb{E}\left[T[r]\right] \quad (25)$$

The above holds for all frames r. Summing over $r \in \{0, 1, \ldots, R-1\}$ for some integer $R > 0$ and using the definition of $\Delta[r]$ gives:

$$\mathbb{E}\left[L[R]\right] - \mathbb{E}\left[L[0]\right] + V\sum_{r=0}^{R-1}\mathbb{E}\left[y_0[r]\right] \leq (B+C)R + Vratio^{opt}\sum_{r=0}^{R-1}\mathbb{E}\left[T[r]\right]$$

Using the fact that $\mathbb{E}\left[L[R]\right] \geq 0$, $\mathbb{E}\left[L[0]\right] = 0$, and rearranging terms yields the result of part (a). □

Proof: (Theorem 1 part (b)) From (25) we have that:

$$\mathbb{E}\left[\Delta[r]\right] \leq B + C + Vratio^{opt}T^{max} - Vy_0^{min} \quad (26)$$

Define constant F as the right-hand-side of the above inequality. Then $\mathbb{E}\left[\Delta[r]\right] \leq F$ for all frames $r \in \{0, 1, 2, \ldots\}$. Further, it can be shown that second moments of queue changes are bounded. Thus, from [15] we have that all queues are rate stable, proving the first part of (b).

Next, summing (26) over $r \in \{0, 1, \ldots, R-1\}$ (for some integer $R > 0$) and using $\Delta[r] = L[r+1] - L[r]$ gives:

$$\mathbb{E}\left[L[R]\right] - \mathbb{E}\left[L[0]\right] \leq [B + C + Vratio^{opt}T^{max} - Vy_0^{min}]R$$

Using the fact that $\mathbb{E}\left[L[0]\right] = 0$ and noting that $L[R] = (1/2)||\boldsymbol{Q}[R]||^2$ gives:

$$\mathbb{E}\left[||\boldsymbol{Q}[R]||^2\right] \leq 2[B + C + V(ratio^{opt}T^{max} - y_0^{min})]R$$

By Jensen's inequality we have $\mathbb{E}\left[||\boldsymbol{Q}[R]||^2\right] \geq ||\mathbb{E}\left[\boldsymbol{Q}[R]\right]||^2$, and so:

$$\mathbb{E}\left[||\boldsymbol{Q}[R]||\right] \leq \sqrt{2[B + C + V(ratio^{opt}T^{max} - y_0^{min})]R}$$

The result follows by dividing the above by R. □

Proof: (Theorem 1 part (c)) First note from (22) that all queues are *mean rate stable*, meaning that for all $l \in \{1, \ldots, L\}$ and $k \in \{1, \ldots, K\}$:

$$\lim_{R\to\infty}\frac{\mathbb{E}\left[Z_l[R]\right]}{R} = \lim_{R\to\infty}\frac{\mathbb{E}\left[|H_k[R]|\right]}{R} = 0 \quad (27)$$

The queue update equation for $Z_l[r]$ in (14) implies that for all frames r:

$$Z_l[r+1] \geq Z_l[r] + y_l[r] - c_l T[r]$$

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

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

Dividing by R and using $Z_l[0] = 0$ gives:

$$\frac{Z_l[R]}{R} \geq \frac{1}{R}\sum_{r=0}^{R-1} y_l[r] - c_l\frac{1}{R}\sum_{r=0}^{R-1} T[r] \quad (28)$$

Taking expectations gives:

$$\frac{\mathbb{E}\left[Z_l[R]\right]}{R} \geq \overline{y}_l[R] - c_l\overline{T}[R]$$

Rearranging terms gives:

$$\overline{y}_l[R]/\overline{T}[R] \leq c_l + \frac{\mathbb{E}\left[Z_l[R]\right]}{T^{min}R} \quad (29)$$

Taking a limit of the above and using (27) proves (23). The corresponding statement where time average expectations are replaced with pure time averages follows from (28) and (20).

The inequality (24) and its pure time average variant can be proven using the queue dynamics (15) (omitted for brevity). □

IV. Markov Implementation

Assume we have a target transition probability matrix $P^* = (P_{ij}^*)$, and target values $y_l^{*(k)}$, $T^{*(k)}$ (where $T^{*(k)} > 0$ for all k). We want to design a policy on the original Markov system (described in the introduction) such that: (i) the fraction of time we transition from i to j is P_{ij}^*, (ii) the conditional average of penalty l, given we are in state k, is at most $y_l^{*(k)}$, and (iii) the conditional average frame size, given we are in state k, is

$T^{*(k)}$. The target values can be those obtained by running the algorithm of the previous section, so that they solve the problem of interest. One might alternatively update these target values in parallel with running the previous algorithm, although we do not explore this approach. Throughout this section, we assume the target values are given and are feasible for the system of interest.

As in the introduction, we define $k[r]$ as the state process over frames $r \in \{0, 1, 2, \ldots\}$. The process $k[r]$ takes values in $\{1, \ldots, K\}$. However, unlike the previous section, $k[r]$ cannot be directly chosen as a decision variable. Rather, it evolves probabilistically according to the transition probabilities $\hat{P}_{ij}(\omega[r], \alpha[r])$. Every frame r, the system controller observes $k[r]$ and $\omega[r]$, and makes a decision $\alpha[r] \in \mathcal{A}(k[r], \omega[r])$. Recall that $y_l[r] = \hat{y}_l(k[r], \omega[r], \alpha[r])$, $T[r] = \hat{T}(k[r], \omega[r], \alpha[r])$, and again define $q_{ij}[r]$ by (7) (with $k[r]$ in this case being the Markov state rather than a control variable). Define $1_k[r]$ as an indicator function that is 1 if $k[r] = k$, and 0 else. We seek to solve the following problem:

$$\begin{aligned}
\text{Minimize:} \quad & \overline{y}_0 && (30)\\
\text{Subject to:} \quad & \overline{y}_l \leq \textstyle\sum_{k=1}^{K} \overline{1}_k y_l^{*(k)} \ \forall l \in \{1, \ldots, L\} && (31)\\
& \overline{T} = \textstyle\sum_{k=1}^{K} \overline{1}_k T^{*(k)} && (32)\\
& \overline{q}_{ij} = \overline{1}_i P_{ij}^* \ \forall i, j \in \{1, \ldots, K\} && (33)\\
& \alpha[r] \in \mathcal{A}(k[r], \omega[r]) \ \forall r \in \{0, 1, 2, \ldots\} && (34)
\end{aligned}$$

where $\overline{1}_k$ is the fraction of time being in state k:

$$\overline{1}_k = \lim_{R\to\infty} \frac{1}{R} \sum_{r=0}^{R-1} \mathbb{E}\left[1_k[r]\right]$$

We note that this is not a standard stochastic network optimization problem of the form [1] because of the Markov dynamics for $k[r]$. In particular, the value of $1_k[r]$ on frame r, which affects the $\overline{1}_k$ value in (31)-(33), is not a decision variable. Rather, it depends on the decisions made in previous frames. The motivation for the problem (30)-(34) is as follows: Suppose the given transition probability matrix P^* yields a system with a unique probability vector $\boldsymbol{\pi}^*$ such that $\boldsymbol{\pi}^* = \boldsymbol{\pi}^* P^*$. Define y_l^* and T^* by:

$$y_l^* = \sum_{k=1}^{K} \pi_k^* y_l^{*(k)} \ \forall l \in \{0, 1, \ldots, L\} \ , \ T^* = \sum_{k=1}^{K} \pi_k^* T^{*(k)}$$

Suppose these values satisfy $y_0^*/T^* = ratio^*$, $y_l^*/T^* \leq c_l$ for all $l \in \{1, \ldots, L\}$, for some desired constants c_l. Then, because the constraint (33) ensures the fraction of time transitioning between states i and j is P_{ij}^*, the fraction of time being in state k is π_k^* (because $\boldsymbol{\pi}^*$ is the unique solution to $\boldsymbol{\pi}^* = \boldsymbol{\pi}^* P^*$). That is, $\overline{1}_k = \pi_k^*$ for all $k \in \{1, \ldots, K\}$. Thus, for all $l \in \{1, \ldots, L\}$ we have by (31) and (32):

$$\frac{\overline{y}_l}{\overline{T}} \leq \frac{\sum_{k=1}^{K} \pi_k^* y_l^{*(k)}}{\overline{T}} = \frac{y_l^*}{T^*} \leq c_l$$

Thus, all desired constraints are satisfied. Furthermore, if we assume y_0^* is achievable, and our policy minimizes $\overline{y}_0$ over all algorithms that meet the constraints, then:

$$\overline{y}_0/\overline{T} = \overline{y}_0/T^* \leq y_0^*/T^* = ratio^*$$

and so the algorithm also achieves (or improves upon) the desired ratio $ratio^*$.

Assumption 2: There exists a (k, ω)-only policy $\alpha^*[r]$, together with a probability distribution $\boldsymbol{\pi}^*$, such that $\boldsymbol{\pi}^* = \boldsymbol{\pi}^* P^*$ and such that for all $l \in \{0, 1, \ldots, L\}$:

$$\begin{aligned}
\mathbb{E}\left[\hat{y}_l(k, \omega[r], \alpha^*[r]) | k[r] = k\right] &\leq y_l^{*(k)} \ \forall k \in \{1, \ldots, K\}\\
\mathbb{E}\left[\hat{T}(k, \omega[r], \alpha^*[r]) | k[r] = k\right] &= T^{*(k)} \ \forall k \in \{1, \ldots, K\}\\
\mathbb{E}\left[\hat{P}_{kj}(\omega[r], \alpha^*[r]) | k[r] = k\right] &= P_{kj}^* \ \forall k, j \in \{1, \ldots, K\}
\end{aligned}$$

A. The Dynamic Algorithm (Algorithm 2)

To enforce the constraints (31)-(33), we define virtual queues $F_l[r]$ for $l \in \{1, \ldots, L\}$, $G[r]$, and $H_{ij}[r]$ for $i, j \in \{1, \ldots, K\}$:

$$\begin{aligned}
F_l[r+1] &= \max[F_l[r] + y_l[r] - \sum_{k=1}^{K} 1_k[r] y_l^{*(k)}, 0] && (35)\\
G[r+1] &= G[r] + T[r] - \sum_{k=1}^{K} 1_k[r] T^{*(k)}[r] && (36)\\
H_{ij}[r+1] &= H_{ij}[r] + q_{ij}[r] - 1_i[r] P_{ij}^* && (37)
\end{aligned}$$

Define $L[r]$ by:

$$L[r] = \frac{1}{2}\sum_{l=1}^{L} F_l[r]^2 + \frac{1}{2} G[r]^2 + \frac{1}{2}\sum_{i=1}^{K}\sum_{j=1}^{K} H_{ij}[r]^2$$

Let $\boldsymbol{Q}[r]$ be the vector of all virtual queue values.

Lemma 2: For any control algorithm, we have:

$$\begin{aligned}
\mathbb{E}\left[\Delta[r] + V y_0[r] | \boldsymbol{Q}[r]\right] &\leq D + V\mathbb{E}\left[y_0[r] | \boldsymbol{Q}[r]\right]\\
&+ \sum_{l=1}^{L} F_l[r]\mathbb{E}\left[y_l[r] - \sum_{k=1}^{K} 1_k[r] y_l^{*(k)} | \boldsymbol{Q}[r]\right]\\
&+ G[r]\mathbb{E}\left[T[r] - \sum_{k=1}^{K} 1_k[r] T^{*(k)}[r] | \boldsymbol{Q}[r]\right]\\
&+ \sum_{i=1}^{K}\sum_{j=1}^{K} H_{ij}[r]\mathbb{E}\left[1_i[r](P_{ij}[r] - P_{ij}^*) | \boldsymbol{Q}[r]\right] \qquad (38)
\end{aligned}$$

where D is a finite constant.

Proof: Omitted for brevity (see [1] for similar results). □

Our dynamic algorithm observes $k[r]$, $\omega[r]$, and queues $\boldsymbol{Q}[r]$ every frame r, and chooses $\alpha[r] \in \mathcal{A}(k[r], \omega[r])$ to minimize the right-hand-side of (38). This reduces to choosing $\alpha[r] \in \mathcal{A}(k[r], \omega[r])$ to deterministically minimize:

$$\begin{aligned}
V\hat{y}_0(k[r], \omega[r], \alpha[r]) &+ \sum_{l=1}^{L} F_l[r]\hat{y}_l(k[r], \omega[r], \alpha[r])\\
&+ G[r]\hat{T}(k[r], \omega[r], \alpha[r])\\
&+ \sum_{j=1}^{K} H_{k[r]j}[r]\hat{P}_{k[r]j}(\omega[r], \alpha[r])
\end{aligned}$$

At the end of each frame, the queues are updated via (35)-(37).

B. Performance of Algorithm 2

Theorem 2: Suppose Assumption 2 holds, that all virtual queues are initially 0, and that $V \geq 0$. If $\omega[r]$ is i.i.d. over frames, then for all frames R:

$$\frac{1}{R}\sum_{r=1}^{R-1}\sum_{k=1}^{K}\mathbb{E}\left[1_k[r](y_0[r]-y_0^{*(k)})\right] \leq D/V \qquad (39)$$

where D is the constant in Lemma 2. Furthermore, for all $l \in \{1,\ldots,L\}$ and $i,j \in \{1,\ldots,K\}$:

$$\lim_{R\to\infty}\frac{1}{R}\sum_{r=1}^{R-1}\sum_{k=1}^{K}\mathbb{E}\left[1_k[r](y_l[r]-y_l^{*(k)})\right] \leq 0 \qquad (40)$$

$$\lim_{R\to\infty}\frac{1}{R}\sum_{r=1}^{R-1}\sum_{k=1}^{K}\mathbb{E}\left[1_k[r](T[r]-T^{*(k)})\right] = 0 \qquad (41)$$

$$\lim_{R\to\infty}\frac{1}{R}\sum_{r=1}^{R-1}\mathbb{E}\left[1_i[r](P_{ij}[r]-P_{ij}^*)\right] = 0 \ \forall i,j \qquad (42)$$

Proof: Because our algorithm minimizes the right-hand-side of (38) over all alternative (possibly randomized) control actions on frame r, we have:

$$\begin{aligned}\mathbb{E}\left[\Delta[r]+Vy_0[r]|\boldsymbol{Q}[r]\right] &\leq D+V\mathbb{E}\left[y_0^*[r]|\boldsymbol{Q}[r]\right]\\ &+\sum_{l=1}^{L}F_l[r]\mathbb{E}\left[y_l^*[r]-\sum_{k=1}^{K}1_k[r]y_l^{*(k)}|\boldsymbol{Q}[r]\right]\\ &+G[r]\mathbb{E}\left[T^*[r]-\sum_{k=1}^{K}1_k[r]T^{*(k)}[r]|\boldsymbol{Q}[r]\right]\\ &+\sum_{i=1}^{K}\sum_{j=1}^{K}H_{ij}[r]\mathbb{E}\left[1_i[r](P_{ij}^*[r]-P_{ij}^*)|\boldsymbol{Q}[r]\right] \qquad (43)\end{aligned}$$

where $y_l^*[r]$, $T^*[r]$, $P_{ij}^*[r]$ correspond to any alternative decisions $\alpha^*[r] \in \mathcal{A}(k[r],\omega[r])$, and are given by:

$$y_l^*[r] = \sum_{k=1}^{K}1_k[r]\hat{y}_l(k,\omega[r],\alpha^*[r]) \qquad (44)$$

$$T^*[r] = \sum_{k=1}^{K}1_k[r]\hat{T}(k,\omega[r],\alpha^*[r]) \qquad (45)$$

$$1_i[r]P_{ij}^*[r] = 1_i[r]\hat{P}_{ij}(\omega[r],\alpha^*[r]) \qquad (46)$$

where we have used the fact that $\sum_{k=1}^{K}1_k[r]=1$ always. Now assume $\alpha^*[r]$ is the (k,ω)-only policy from Assumption 2, which is independent of $\boldsymbol{Q}[r]$. Taking expectations of (44)-(46) and using Assumption 2 gives for all $l \in \{0,1,\ldots,L\}$ and all $i,j \in \{1,\ldots,K\}$:

$$\begin{aligned}\mathbb{E}\left[y_l^*[r]|\boldsymbol{Q}[r]\right] &\leq \sum_{k=1}^{K}\mathbb{E}\left[1_k[r]|\boldsymbol{Q}[r]\right]y_l^{*(k)}\\ \mathbb{E}\left[T^*[r]|\boldsymbol{Q}[r]\right] &= \sum_{k=1}^{K}\mathbb{E}\left[1_k[r]|\boldsymbol{Q}[r]\right]T^{*(k)}\\ \mathbb{E}\left[1_i[r]P_{ij}^*[r]|\boldsymbol{Q}[r]\right] &= \mathbb{E}\left[1_i[r]|\boldsymbol{Q}[r]\right]P_{ij}^*\end{aligned}$$

Using the above in (43) gives:

$$\mathbb{E}\left[\Delta[r]+Vy_0[r]|\boldsymbol{Q}[r]\right] \leq D+V\sum_{k=1}^{K}\mathbb{E}\left[1_k[r]|\boldsymbol{Q}[r]\right]y_0^{*(k)}$$

Rearranging terms and again using $\sum_{k=1}^{K}1_k[r]=1$ gives:

$$\mathbb{E}\left[\Delta[r]+V\sum_{k=1}^{K}1_k[r](y_0[r]-y_0^{*(k)})|\boldsymbol{Q}[r]\right] \leq D$$

Taking expectations of the above (with respect to $\boldsymbol{Q}[r]$) and using iterated expectations gives:

$$\mathbb{E}\left[\Delta[r]\right]+V\sum_{k=1}^{K}\mathbb{E}\left[1_k[r](y_0[r]-y_0^{*(k)})\right] \leq D \qquad (47)$$

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

$$\mathbb{E}\left[L[R]\right]-\mathbb{E}\left[L[0]\right]+V\sum_{r=0}^{R-1}\sum_{k=1}^{K}\mathbb{E}\left[1_k[r](y_0[r]-y_0^{*(k)})\right] \leq D$$

Rearranging terms and using $\mathbb{E}\left[L[R]\right] \geq 0$ and $\mathbb{E}\left[L[0]\right]=0$ proves (39).

Now note that (47) implies that for all frames $r \in \{0,1,2,\ldots\}$:

$$\mathbb{E}\left[\Delta[r]\right] \leq F$$

for some finite constant F. As in Theorem 1, this implies all queues are rate stable and mean rate stable, so that all constraints (40)-(42) hold. □

REFERENCES

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

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

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

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

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

[6] 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*, Dec. 2003.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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, Dec. 2009.

[12] 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.

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

[14] 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.

[15] M. J. Neely. Queue stability and probability 1 convergence via lyapunov optimization. *Arxiv Technical Report*, Oct. 2010.