Delay-Based Network Utility Maximization

Michael J. Neely, *Senior Member, IEEE*

***Abstract*—It is well known that max-weight policies based on a queue backlog index can be used to stabilize stochastic networks, and that similar stability results hold if a delay index is used. Using Lyapunov optimization, we extend this analysis to design a utility maximizing algorithm that uses explicit delay information from the head-of-line packet at each user. The resulting policy is shown to ensure deterministic worst-case delay guarantees and to yield a throughput utility that differs from the optimally fair value by an amount that is inversely proportional to the delay guarantee. Our results hold for a general class of 1-hop networks, including packet switches and multiuser wireless systems with time-varying reliability.**

***Index Terms*—Optimization, queueing, stochastic control.**

I. Introduction

THIS paper considers the problem of scheduling for maximum throughput utility in a network with random packet arrivals and time-varying channel reliability. We focus on 1-hop networks where each packet requires transmission over only one link. At every slot, the network controller assesses the condition of its channels and selects a set of links for transmission. The success of each transmission depends on the collection of links selected and their corresponding reliabilities. The goal is to maximize a concave and nondecreasing function of the time-average throughput on each link. Such a function represents a *utility function* that acts as a measure of fairness for the achieved throughput vector.

In the case when traffic is inside the *network capacity region*, the utility-optimal throughput vector is simply the vector of arrival rates, and the problem reduces to a *network stability problem*. In this case, it is well known that the network can be stabilized by *max-weight* policies that schedule links every slot to maximize a weighted sum of transmission rates, where the weights are queue backlogs. This is typically shown via a Lyapunov drift argument (see [2] and references therein). This technique for stable control of a queueing network was first used for link and server scheduling in [3] and [4] and has since become a powerful method to treat stability in different contexts, including switches and computer networks [5]–[7], wireless systems and ad hoc mobile networks with rate and power allocation [8]–[10], and systems with probabilistic channel errors [11], [12].

Manuscript received May 25, 2011; revised December 20, 2011; accepted February 22, 2012; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor I. Keslassy. Date of publication April 09, 2012; date of current version February 12, 2013. This work was supported by the DARPA IT-MANET under Grant W911NF-07-0028, the NSF under Career Grant CCF-0747525, and the Network Science Collaborative Technology Alliance sponsored by the US Army Research Laboratory W911NF-09-2-0053. This work was presented in part at the IEEE International Conference on Computer Communications (INFOCOM), San Diego, CA, March 15–19, 2010.

The author is with the University of Southern California, Los Angeles, CA 90089 USA (e-mail: mjneely@usc.edu).

Digital Object Identifier 10.1109/TNET.2012.2191157

In the case when traffic is either inside or outside of the capacity region, it is known that the max-weight policy can be combined with a flow control policy to jointly stabilize the network while maximizing throughput utility. This is shown in [2], [13], and [14] via a Lyapunov optimization argument, and in [15] via a fluid limit analysis. Utility optimization for the special case of "infinitely backlogged" sources is shown in [16]–[18] and was perhaps first addressed for time-varying wireless downlinks without explicit queueing in [19] and [20].

The stability works [3]–[12] all use backlog-based transmission rules, as do the works in [2] and [13]–[18], which treat joint stability and utility optimization. However, work in [21] introduces an interesting *delay-based* Lyapunov function for proving stability, where the delay of the head-of-line packet is used as a weight in the max-weight decision. This approach intuitively provides tighter control of the actual queueing delays. Indeed, a single head-of-line packet is scheduled based on the delay it has experienced, rather than on the amount of additional packets that arrived after it. This delay-based approach to queue stability is extended in [22], where the *Modified Largest Weighted Delay First* algorithm is developed, and in [23], which uses a delay-based exponential rule. However, [21]–[23] use delay-based rules only in the context of queue stability. To our knowledge, there are no prior works that use delay-based scheduling to address the important issue of joint stability and utility optimization.

This paper fills that gap. We use a delay-based Lyapunov function and extend the analysis to treat joint stability and performance optimization via the Lyapunov optimization technique from our prior work [2], [13], [14]. The extension is not obvious. Indeed, the flow control decisions in the prior work [2], [13], [14] are made immediately when a new packet arrives, which directly affects the drift of backlog-based Lyapunov functions. However, such decisions *do not* directly affect the delay value of the head-of-line packets, and hence do not directly affect the drift of delay-based Lyapunov functions. We overcome this challenge with a novel flow control policy that queues *all* arriving data, but makes packet dropping decisions just before advancing a new packet to the head-of-line. This policy is structurally different from the utility optimization works [2] and [13]–[20]. This new structure leads to deterministic guarantees on the worst-case delay of any nondropped packet and provides throughput utility that can be pushed arbitrarily close to optimal. Specifically, for any integer $D \geq 2$, we can construct an algorithm that ensures all nondropped packets have delay less than or equal to D slots, with total throughput utility that differs from optimal by $O(1/D)$. The deterministic delay guarantee is particularly challenging to establish, and for this we introduce a new technique of *concavely extending a utility function*.

Similar $[O(1/D), O(D)]$ performance tradeoffs are shown for queue-based Lyapunov functions in the previous work [2], [13], [14] (see also [24]–[26] for improved tradeoffs),

but these guarantees apply only to queue size, rather than delay.[1] The deterministic delay guarantees we obtain in this present paper are quite strong and show the advantages of our new flow control structure. However, a disadvantage is that admit/drop decisions are delayed until a packet is at the head-of-line, rather than being determined immediately upon arrival. Moreover, due to correlation issues unique to this delay-based scenario, analysis is simplified if we assume the scheduler knows the vector of arrival rates to each link (although we also generalize to cases when these rates are unknown). Furthermore, while our deterministic delay guarantees hold for general arrival sample paths, our utility analysis assumes all arrival processes are independent of each other (possibly with different rates for each process) and independent and identically distributed (i.i.d.) over time-slots. Nevertheless, it is important to analyze these delay-based policies because they improve our understanding of network delay, and because the deterministic guarantees they offer are useful for many practical systems.

We further show via simulation that our algorithms maintain good performance when the i.i.d. arrivals are replaced by ergodic but temporally correlated "bursty" arrivals with the same rates. However, the worst-case delay required to achieve the same utility performance is increased in this case. This is not surprising if we compare to known results for backlog-based Lyapunov algorithms. Backlog-based algorithms were first developed under i.i.d. assumptions, but later shown to work—with increased delay—for non-i.i.d. cases (see [28] and references therein). Thus, while we limit our analytical proofs to the i.i.d. setting, we expect the algorithm to approach optimal utility in more general cases, as supported by our simulations.

While our algorithm can be used to enforce any desired delay guarantee, it is important to emphasize that it does *not* maximize throughput utility subject to this guarantee. Such a problem can be addressed with Markov decision theory, which brings with it the *curse of dimensionality* (see structural results and approximations in [29] and weighted stochastic shortest-path approaches in [30]). In this paper, we claim only that the achieved utility is within $O(1/D)$ of the largest possible utility of any stabilizing algorithm. However, because (for large D) our utility is close to this ideal utility value, it is even closer to the maximum utility that can be achieved subject to the worst-case delay constraint. That is because a basic stability constraint is less stringent than a worst-case delay constraint, and so the optimal utility under a stability constraint is greater than or equal to the optimal utility under a worst-case delay constraint. Furthermore, our approach offers the low-complexity advantages associated with Lyapunov drift and Lyapunov optimization. Specifically, the policy makes real-time transmission decisions based only on the current system state and does not require *a priori* knowledge of the channel-state probabilities. The flow control decisions here can also be implemented in a distributed fashion at each link, as is the case with most other Lyapunov-based utility optimization algorithms.

It is important to distinguish our work, which considers *actual network delay*, from work that approximates network delay as a convex function of a flow rate (such as in [27] and [31]). While it is known that average queue congestion and delay is convex in the arrival rate if traffic from an arbitrary arrival process is probabilistically split [32], this is not necessarily true (or relevant) for dynamically controlled networks, particularly when the control depends on the queue backlogs and delays themselves. Actual network delay problems involve not only optimization of rate based utility functions, but engineering of the Lagrange multipliers (which are related to queue backlogs) associated with those utility functions [25], [26].

Prior work on throughput-optimal control in networks with finite buffers is in [33]–[35], where [33] considers queue stability, [34] considers maximum throughput subject to average power constraints, and [35] considers utility-optimal flow control. Work in [36] considers hop-count constrained scheduling. The works [33]–[36] all treat multihop systems, but use backlog-based scheduling and do not provide worst-case delay guarantees. Our recent conference paper [37], developed as an extension of this current paper, treats worst-case delay guarantees in multihop networks in considerably more general scenarios than the current paper. It treats non-i.i.d. and nonergodic systems for which multiple packets can arrive and depart a single queue each slot. However, it requires use of an *ϵ-persistent service queue* for a carefully chosen $\epsilon > 0$. Achieved utility can degrade if ϵ is too large, and the delay bound grows proportionally to V/ϵ, where V is another parameter that affects proximity to optimal utility. The current paper considers a more restrictive setting, but obtains tighter results that do not use an ϵ parameter.

II. Network Model

Consider a 1-hop network that operates in discrete time with normalized time-slots $t \in \{0, 1, 2, \ldots\}$. There are L links, and packets arrive randomly every slot and are queued separately for transmission over each link. Let $\boldsymbol{A}(t) = (A_1(t), \ldots, A_L(t))$ be the process of random packet arrivals, where $A_l(t)$ is the number of packets that arrive to link l on slot t. For simplicity, assume that all packets have fixed size, and that there is at most one packet arrival to each link per slot, so that $A_l(t) \in \{0, 1\}$ for all links l and slots t. The arrival vector $\boldsymbol{A}(t)$ is assumed to be i.i.d. over slots. Furthermore, the arrival processes $A_l(t)$ for different links in each slot are assumed to be independent. Let $\boldsymbol{Q}(t) = (Q_1(t), \ldots, Q_L(t))$ denote the integer number of packets currently stored in each of the L queues. All packets are marked with their integer arrival slot, which is used to determine their delay in the system. The one-step queueing equation for each link l is

$$Q_l(t+1) = \max\left[Q_l(t) - \mu_l(t) - D_l(t), 0\right] + A_l(t) \quad (1)$$

where $\mu_l(t)$ represents the amount of packets successfully served on slot t, and $D_l(t)$ represents the number of packets *dropped* on slot t. A packet can be dropped at any time, although in our specific algorithm we impose a two-stage structure that first makes a transmission decision and then makes a dropping decision in reaction to the feedback obtained from the transmission.

A. Time-Varying Link Reliability

For simplicity, assume that each link can transmit at most one packet per slot, so that $\mu_l(t) \in \{0, 1\}$ for all links l and all slots t. Let $\boldsymbol{x}(t) = (x_1(t), \ldots, x_L(t))$ denote a *transmission*

[1]Of course *average* delay and *average* backlog are directly related through Little's Theorem [27], but this is not true for worst-case backlog and delay.

vector, where $x_l(t) \in \{0, 1\}$, and $x_l(t) = 1$ if link l attempts transmission on slot t. Let $\mathcal{X}$ denote the set of all allowable link transmission vectors, possibly being the set of all 2^L such vectors, but also possibly incorporating some constraints (such as permutation constraints for $N \times N$ packet switches). We assume $\mathcal{X}$ has the natural property that for any $\boldsymbol{x} \in \mathcal{X}$, all subvectors $\boldsymbol{x}'$, formed by setting one or more entries of $\boldsymbol{x}$ to zero, are also in $\mathcal{X}$. It is useful to assume a link can transmit even if it does not have a packet, in which case a *null packet* is transmitted. Let $\boldsymbol{S}(t) = (S_1(t), \ldots, S_L(t))$ denote a link condition vector for slot t, which determines the probability of successful transmission on each slot. Specifically, given particular $\boldsymbol{x}(t)$ and $\boldsymbol{S}(t)$ vectors, the probability of successful transmission on link l is given by a *reliability function*

$$\Pr\left[\text{link } l \text{ success}|\boldsymbol{x}(t), \boldsymbol{S}(t)\right] = \Psi_l\left(\boldsymbol{x}(t), \boldsymbol{S}(t)\right). \quad (2)$$

The reliability function $\Psi_l(\boldsymbol{x}, \boldsymbol{S})$ for each $l \in \{1, \ldots, L\}$ is general and is assumed only to take real values between 0 and 1 (representing probabilities), and to have the property that $\Psi_l(\boldsymbol{x}, \boldsymbol{S}) = 0$ whenever $x_l = 0$. The channel condition vector $\boldsymbol{S}(t)$ is assumed to be i.i.d. over slots and independent of the $\boldsymbol{A}(t)$ process. Assume $\boldsymbol{S}(t)$ takes values in a set $\mathcal{S}$ of arbitrary cardinality. The vector $\boldsymbol{S}(t)$ is known to the network controller at the beginning of each slot t. In practice, $\boldsymbol{S}(t)$ is the result of a channel measurement or estimation that is done every slot. The estimate might be inexact, in which case the reliability function $\Psi_l(\boldsymbol{x}(t), \boldsymbol{S}(t))$ represents the probability that the actual network channels on slot t are sufficient to support the attempted transmission over link l (given $\boldsymbol{x}(t)$ and the estimate $\boldsymbol{S}(t)$ for slot t).

We assume the reliability function is known. Recent online techniques for estimation of packet error rates are considered in [38]. In the context of [38], a number of other decision parameters to be chosen on each slot also affect reliability, such as modulation, power levels, subband selection, coding type, etc. These choices can be represented as a parameter space $\mathcal{I}$. In this case, the reliability function can be extended to include the parameter choice $I(t) \in \mathcal{I}$ made every slot: $\Psi_l(\boldsymbol{x}(t), \boldsymbol{S}(t), I(t))$. This does not change our mathematical analysis (see also Remark 1 in Section III-F), although for simplicity we focus on the reliability function structure of (2).

We assume that ACK/NACK information is given at the end of the slot to inform each link if its transmission was successful or not. Packets that are not successful do not leave the queue (unless they are dropped in a packet drop decision). With this model of link success, the transmission variable $\mu_l(t)$ in (1) is given by

$$\mu_l(t) = x_l(t) 1_l(t)$$

where $1_l(t)$ is an indicator variable that is 1 if the transmission over link l is successful, and 0 otherwise. That is

$$1_l(t) = \begin{cases} 1, & \text{with probability } \Psi_l\left(\boldsymbol{x}(t), \boldsymbol{S}(t)\right) \\ 0, & \text{with probability } 1 - \Psi_l\left(\boldsymbol{x}(t), \boldsymbol{S}(t)\right). \end{cases}$$

The successes/failures over each link on slot t are assumed to be independent of past events given the current $\boldsymbol{x}(t)$ and $\boldsymbol{S}(t)$ values. The successes/failures might be correlated over each link. This is not captured in the $\Psi_l(\boldsymbol{x}, \boldsymbol{S})$ functions alone and can only be fully described by a joint success distribution function for all 2^L possible success/failure outcomes for a given $\boldsymbol{x}$ and $\boldsymbol{S}$. However, it turns out that the network capacity region, and hence the associated maximum utility point, is independent of such interlink success correlations [12]. Hence, it suffices to use only the *marginal* distribution functions $\Psi_l(\boldsymbol{x}, \boldsymbol{S})$ for each $l \in \{1, \ldots, L\}$.

B. Examples of Packet Switches and Wireless Networks

The above model applies to a wide class of 1-hop networks. For example, it applies to the $N \times N$ packet switch models of [5] and [7] by defining $\boldsymbol{S}(t)$ to be a null vector (so that there is no notion of channel variation) and by defining $\mathcal{X}$ as the set of all link transmission vectors that satisfy permutation constraints (see Section VI-A). For wireless networks with interference but without time-varying channels, the set $\mathcal{X}$ can be defined as all link activations that do not interfere with each other (i.e., that do not produce *collisions*), as in [3]. The reliability function $\Psi_l(\cdot)$ can be used to extend the model to treat cases where interfering links result in probabilistic reception.

Furthermore, the opportunistic scheduling systems of [4] with time-varying ON/OFF channels can be modeled with $\boldsymbol{S}(t)$ being the vector of ON/OFF channel states on each slot, and with the function $\Psi_l(\boldsymbol{x}, \boldsymbol{S})$ taking the value 1 whenever $x_l = 1$ and $S_l =$ ON, and 0 otherwise. Finally, the model supports probabilistic reception in the case when the link reliability can vary from slot to slot.

A simple example is when $S_l(t)$ represents the current probability that a link l transmission would be successful, so that

$$\Psi_l\left(\boldsymbol{x}(t), \boldsymbol{S}(t)\right) = \begin{cases} S_l(t), & \text{if } x_l(t) = 1 \\ 0, & \text{if } x_l(t) = 0. \end{cases}$$

This example has the success probability over link l a pure function of $x_l(t)$ and $S_l(t)$, and hence implicitly assumes that the set $\mathcal{X}$ limits all simultaneous link transmissions to orthogonal channels. More complex interchannel interference models can be described by more complex $\Psi_l(\boldsymbol{x}, \boldsymbol{S})$ functions.

III. Delay-Based Flow Control

Let $\boldsymbol{\lambda} = \mathbb{E}[\boldsymbol{A}(t)]$ be the vector of arrival rates, so that $\lambda_l = \mathbb{E}[A_l(t)]$ is the arrival rate to link l (in units of packets/slot). The *network capacity region* Λ is defined as the closure of the set of all long-term throughput vectors that the system can support. The set Λ is known to be the same as the closure of the set of all arrival rate vectors $\boldsymbol{\lambda}$ for which there exists a stabilizing scheduling algorithm, subject to the constraint that *the flow controllers are turned off* (so that no packets are dropped and $D_l(t) = 0$ for all l and all t) [4], [12]. Specifically, in [12] it is shown that the set Λ is given by the set of all time-average transmission rates that can be achieved by stationary and randomized algorithms, called $\boldsymbol{S}$-only algorithms, that observe $\boldsymbol{S}(t)$ every slot t and choose a (possibly random) transmission vector $\boldsymbol{x}(t) \in \mathcal{X}$ according to a probability distribution that depends only on the observed channel state $\boldsymbol{S}(t)$. Thus, for every vector $\boldsymbol{r} \in \Lambda$, with $\boldsymbol{r} = (r_1, \ldots, r_L)$, there is an $\boldsymbol{S}$-only algorithm $\boldsymbol{x}^*(t)$, with a corresponding random service vector $\boldsymbol{\mu}^*(t) = (x_1^*(t)1_1^*(t), \ldots, x_L^*(t)1_L^*(t))$ that yields for each $l \in \{1, \ldots, L\}$

$$r_l = \mathbb{E}\left[\mu_l^*(t)\right] = \mathbb{E}\left[x_l^*(t)\Psi_l\left(\boldsymbol{x}^*(t), \boldsymbol{S}(t)\right)\right] \quad (3)$$

where the expectation in (3) is with respect to the distribution of $\boldsymbol{S}(t)$ and the distribution of $\boldsymbol{x}^*(t)$ given $\boldsymbol{S}(t)$.

A. Optimization Objective

Let $g(\boldsymbol{y})$ be a continuous and concave utility function of the L-dimensional vector $\boldsymbol{y} = (y_1, \ldots, y_L)$, where $\boldsymbol{y}$ is used to represent the time-average throughput on each link (in units of packets/slot). The function can take positive or negative values and is assumed to be defined over the hyper-cube $\mathbf{0} \leq \boldsymbol{y} \leq \mathbf{1}$, where inequality is taken entrywise, and $\mathbf{0}$ and $\mathbf{1}$ are vectors with all entries equal to 0 and 1, respectively. Assume that $g(\boldsymbol{y})$ is nondecreasing in each entry y_l. An example is the *separable* utility function

$$g(\boldsymbol{y}) = \sum_{l=1}^{L} g_l(y_l) \tag{4}$$

where for each link l, $g_l(y_l)$ is a concave and nondecreasing function of y_l, defined over the interval $0 \leq y_l \leq 1$. We make the following additional assumption.

Assumption 1: For each $l \in \{1, \ldots, L\}$, the lth right partial derivative of $g(\boldsymbol{y})$, over all $\boldsymbol{y} \in [0,1]^L$ such that $y_l < 1$, is bounded above by a finite constant ν_l, where $\nu_l \geq 0$.[2]

Assumption 1 implies that for any vectors $\boldsymbol{y}$ and $\boldsymbol{w}$ such that $\boldsymbol{y} \in [0,1]^L$, $\boldsymbol{w} \in [0,1]^L$, and $\boldsymbol{y} + \boldsymbol{w} \in [0,1]^L$, we have

$$g(\boldsymbol{y} + \boldsymbol{w}) \leq g(\boldsymbol{y}) + \sum_{l=1}^{L} \nu_l w_l. \tag{5}$$

Note that Assumption 1 does *not* hold for the logarithmic utility function $\sum_l \log(y_l)$ associated with proportional fairness [39]. However, it *does* hold for the following useful utility function example, which is often a good alternative way to treat network fairness:

$$g(\boldsymbol{y}) = \sum_{l} \log(1 + \nu_l y_l). \tag{6}$$

The above is also an approximation of proportional fairness when $\nu_l = \nu$ for all l, for some large value ν.

For each link l, define $Y_l(t)$ as

$$Y_l(t) \triangleq \lambda_l - D_l(t). \tag{7}$$

Let $\overline{\boldsymbol{y}}(t)$ be the time-average expectation of $Y_l(t)$ over t slots

$$\overline{\boldsymbol{y}}(t) \triangleq \boldsymbol{\lambda} - \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\left[\boldsymbol{D}(\tau)\right] \tag{8}$$

where $\boldsymbol{D}(\tau) = (D_1(\tau), \ldots, D_L(\tau))$ is the drop vector for slot τ. Let $\overline{\boldsymbol{y}}$ denote the limit of $\overline{\boldsymbol{y}}(t)$ as $t \to \infty$ (temporarily assume the limit exists). The vector $\overline{\boldsymbol{y}}$ is the difference between the rate of arrivals and packet drops, and hence (if queues are stable) represents the throughput vector. The goal is to design a delay-based transmission scheme with packet dropping that solves the following problem:

$$\begin{aligned} \text{Maximize :} \quad & g(\overline{\boldsymbol{y}}) && (9)\\ \text{Subject to :} \quad & \overline{\boldsymbol{y}} \in \Lambda && (10)\\ & 0 \leq \overline{y}_l \leq \lambda_l \qquad \text{for all } l \in \{1, \ldots, L\}. && (11) \end{aligned}$$

[2]Right partial derivatives exist for any concave function, including nondifferentiable functions such as $g(\boldsymbol{y}) = \min[y_1, \ldots, y_L]$, for which $\nu_l = 1$ for all $l \in \{1, \ldots, L\}$.

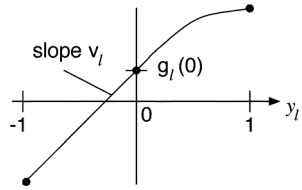


Fig. 1. Illustration of the concave extension of $g_l(y_l)$.

Let g^* be the supremum utility value for the above problem. In addition to striving to achieve a utility that is close to g^*, we desire the actual delays of nondropped packets to be deterministically bounded.

B. Concavely Extended Utility Function

Suppose $g(\boldsymbol{y})$ satisfies Assumption 1. Define the *concave extension of* $g(\boldsymbol{y})$ as the function $\hat{g}(\boldsymbol{y})$ defined over all $\boldsymbol{y} = (y_1, \ldots, y_L)$ such that $-1 \leq y_l < \infty$ for all $l \in \{1, \ldots, L\}$

$$\hat{g}(\boldsymbol{y}) \triangleq g\left([\boldsymbol{y}]_0^1\right) + \sum_{l=1}^{L} \nu_l \min[y_l, 0] \tag{12}$$

where $[\boldsymbol{y}]_0^1$ represents an entrywise projection to interval [0, 1]

$$[\boldsymbol{y}]_0^1 \triangleq \left(\min\left[\max[y_1, 0], 1\right], \ldots, \min\left[\max[y_L, 0], 1\right]\right).$$

Clearly, $\hat{g}(\boldsymbol{y})$ is entrywise nondecreasing, and

$$g(\boldsymbol{y}) = \hat{g}(\boldsymbol{y}) \qquad \text{whenever } \mathbf{0} \leq \boldsymbol{y} \leq \mathbf{1}.$$

It can be shown that $\hat{g}(\boldsymbol{y})$ is concave over the region of all $(y_1, \ldots, y_L)$ such that $-1 \leq y_l < \infty$ for all $l \in \{1, \ldots, L\}$. Furthermore, because (5) holds, it can be shown that for any vector $\boldsymbol{y}$ in this region and any index $l \in \{1, \ldots, L\}$, we have

$$\hat{g}(\boldsymbol{y}) \leq \hat{g}\left(\boldsymbol{y}_l'\right) + \nu_l(y_l + 1) \tag{13}$$

where the vector $\boldsymbol{y}_l'$ is formed from the vector $\boldsymbol{y}$ by replacing the single entry y_l with -1.

In the case when $g(\boldsymbol{y})$ has the separable form (4), the concave extension is given by $\hat{g}(\boldsymbol{y}) = \sum_l \hat{g}_l(y_l)$, where each function $\hat{g}_l(y_l)$ concavely extends the function $g_l(y_l)$, originally defined over the interval $0 \leq y_l \leq 1$, to the interval $-1 \leq y_l < \infty$, as shown in Fig. 1. This method of concavely extending the utility function is crucial to engineer the network delays to be bounded [in particular, it is needed to allow $\gamma_l(t) = -1$ in (28)].

C. Problem Transformation With Virtual Queues

It is not difficult to show that the stochastic network optimization problem (9)–(11) can be transformed using a vector $\boldsymbol{\gamma}(t) = (\gamma_1(t), \ldots, \gamma_L(t))$ of *auxiliary variables* that are chosen every slot according to the constraints $-1 \leq \gamma_l(t) \leq 1$. The transformed problem is

$$\begin{aligned} \text{Maximize :} \quad & \hat{g}(\overline{\boldsymbol{\gamma}}) && (14)\\ \text{Subject to :} \quad & \overline{Q}_l < \infty \qquad \text{for all } l \in \{1, \ldots, L\} && (15)\\ & \overline{y}_l \geq \overline{\gamma}_l \qquad \text{for all } l \in \{1, \ldots, L\} && (16)\\ & -1 \leq \overline{\gamma}_l \leq 1 \qquad \text{for all } l \in \{1, \ldots, L\} && (17)\\ & \overline{\boldsymbol{Q}} \text{ and } \overline{\boldsymbol{y}} \text{ are achievable on the network} && (18) \end{aligned}$$

where $\overline{Q}_l$ is defined

$$\overline{Q}_l \triangleq \limsup_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\left[Q_l(\tau)\right].$$

We say that a nonnegative discrete-time stochastic process $Q_l(t)$ is *strongly stable* if $\overline{Q}_l < \infty$.

This transformation can be intuitively understood as follows. The constraint (11) automatically holds for any achievable control policy, as the throughput cannot be larger than the raw arrival rate, and hence is satisfied whenever (18) holds. The constraint (10) is ensured by the stability constraint (15) in the transformed problem. Finally, one can always choose the auxiliary vector $\boldsymbol{\gamma}(t) = \boldsymbol{y}(t)$ to ensure that (16) and (17) are satisfied (note that $0 \leq \overline{y}_l \leq 1$ for all l because arrival rates cannot be larger than 1). The fact that $g(\boldsymbol{y})$ is nondecreasing in each entry and that $\hat{g}(\boldsymbol{y}) = g(\boldsymbol{y})$ whenever $\mathbf{0} \leq \boldsymbol{y} \leq \mathbf{1}$ ensures that it suffices to consider all constraints (16) holding with equality, so that any control algorithm that solves (14)–(18) also solves (9)–(11).

The auxiliary variables $\boldsymbol{\gamma}(t)$ are important for solving problems of maximizing a concave function of a time average and are crucial for network utility maximization with randomly arriving traffic [2], [28]. To ensure that the constraints (16) are satisfied, we use a *virtual queue* $Z_l(t)$ for each link l, with update equation as follows:

$$Z_l(t+1) = \max\left[Z_l(t) - \lambda_l + D_l(t) + \gamma_l(t), 0\right]. \quad (19)$$

Stabilizing this virtual queue ensures that the time-average value of $Y_l(t)$, defined in (7), is greater than or equal to the time average of $\boldsymbol{\gamma}_l(t)$, which ensures (16). Specifically, using the definition $Y_l(t) = \lambda_l - D_l(t)$, from (19) it is clear that

$$\begin{aligned} Z_l(t+1) &= \max\left[Z_l(t) - Y_l(t) + \gamma_l(t), 0\right] \\ &\geq Z_l(t) - Y_l(t) + \gamma_l(t) \end{aligned}$$

and hence (by summing the above over $\tau \in \{0, \ldots, t-1\}$ and dividing by t)

$$\frac{Z_l(t) - Z_l(0)}{t} + \frac{1}{t}\sum_{\tau=0}^{t-1} Y_l(\tau) \geq \frac{1}{t}\sum_{\tau=0}^{t-1}\gamma_l(\tau).$$

Taking expectations of both sides and using $Z_l(0) = 0$ yields

$$\frac{\mathbb{E}\left[Z_l(t)\right]}{t} + \overline{y}_l(t) \geq \overline{\gamma}_l(t) \quad (20)$$

where $\overline{y}_l(t)$ is the time-average expected value of $Y_l(\tau)$, defined in (8), and $\overline{\gamma}_l(t)$ is defined similarly. It follows from (20) that if $\mathbb{E}[Z_l(t)]/t \to 0$ (a property that is satisfied if $Z_l(t)$ is strongly stable, as shown in [28]), and if $\overline{y}_l(t)$ and $\overline{\gamma}_l(t)$ have well-defined limits $\overline{y}_l$ and $\overline{\gamma}_l$, then $\overline{y}_l \geq \overline{\gamma}_l$, so that the constraints (16) are satisfied.

Implementation of the virtual queue (19) assumes the arrival rates λ_l are known for each link l. For simplicity, we first analyze this case. However, if the rates λ_l are unknown, one can modify the virtual queue update rule (19) to

$$Z_l(t+1) = \max\left[Z_l(t) - A_l(t-W) + D_l(t) + \gamma_l(t), 0\right]$$

where W is a suitably large positive integer. This modification is analyzed in Section V. The use of a W-shifted arrival process $A_l(t-W)$ is unique to this delay-based analysis and is required to handle subtle correlation issues between packet interarrival times and virtual queue states.

D. Delay-Based Lyapunov Function

We now impose the following structure on our control policy. Every slot, a packet transmission decision is made *first*. If a transmission over link l is successful (so that $\mu_l(t) = 1$), then the packet is removed from the queue, and no packet is dropped from link l (so that $D_l(t) = 0$). Else, if link l either did not attempt transmission or if its transmission was unsuccessful, we can decide whether or not to drop the packet, but no other packet can be dropped from link l. Thus, every slot t, we have $0 \leq \mu_l(t) + D_l(t) \leq 1$. We show later that this structure does not hinder our maximum utility objective. Furthermore, it is useful to consider the possibility of transmitting or dropping a *null packet* when the queue is empty, so that $\mu_l(t)$ and $D_l(t)$ in principle can be chosen independently of queue backlog.

Let $H_l(t)$ represent the waiting time of the head-of-line packet in link l on slot t (being at least one if there is a packet), and define $H_l(t) = 0$ if there are no packets in link l at this time. A new packet that arrives to an empty queue on slot t is not placed to the head-of-line until the next slot and is designated to have a waiting time of 1 at slot $t+1$. Define $\alpha_l(t)$ as an indicator variable that is 1 if $Q_l(t) > 0$, and is zero if the queue is empty. Let $\beta_l(t) = 1 - \alpha_l(t)$. Similar to [21], we observe that $H_l(t)$ satisfies the following:

$$\begin{aligned} H_l(t+1) = \alpha_l(t)\max\left[H_l(t) + 1 - (\mu_l(t) + D_l(t))\,T_l(t), 0\right] \\ + \beta_l(t)A_l(t) \end{aligned} \quad (21)$$

where $T_l(t)$ represents the *interarrival time* between the head-of-line packet and the subsequent packet (possibly unknown to the network controller if the subsequent packet has not yet arrived). Because arrivals are Bernoulli, if $H_l(t) > 0$, then $T_l(t)$ is a geometric random variable with success probability λ_l, and $T_l(t)$ takes values in the set $\{1, 2, 3, \ldots\}$. If $H_l(t) = 0$, then we define $T_l(t) = 0$.

Equation (21) can be understood as follows. If $\alpha_l(t) = 0$, then $\beta_l(t) = 1$ so that queue l is empty. In this case, the value of $H_l(t+1)$ is 1 if and only if there is a new arrival on slot t. Alternatively, if $\alpha_l(t) = 1$, then $\beta_l(t) = 0$. Suppose in this case that the head-of-line packet is neither served nor dropped (so that $\mu_l(t) + D_l(t) = 0$). Then, its delay increases by 1, as described by (21). On the other hand, if the head-of-line packet is either served or dropped (so that $\mu_l(t) + D_l(t) = 1$), then the *next* packet enters the head-of-line, with a total waiting time equal to $H_l(t) + 1 - T_l(t)$ (where the additional "+1" comes because this operation takes one more slot). The $\max[\cdot, 0]$ captures the possibility that the interarrival time is greater than $H_l(t) + 1$, in which case the queue is empty on slot $t+1$ with $H_l(t+1) = 0$.

Without loss of generality, assume that $\lambda_l > 0$ for all links $l \in \{1, \ldots, L\}$ (else, just remove the links that have no traffic). Define $\boldsymbol{\Theta}(t) \triangleq [\boldsymbol{Z}(t); \boldsymbol{H}(t)]$, where $\boldsymbol{Z}(t)$ and $\boldsymbol{H}(t)$ are vectors of the virtual queues in (19) and the head-of-line values in (21). We use the following nonnegative Lyapunov function:

$$L\left(\boldsymbol{\Theta}(t)\right) \triangleq \frac{1}{2}\sum_{l=1}^{L} Z_l(t)^2 + \frac{1}{2}\sum_{l=1}^{L}\lambda_l H_l(t)^2.$$

E. Minimizing the Drift-Plus-Penalty

Define $\Delta(\boldsymbol{\Theta}(t))$ as the *one-step conditional Lyapunov drift*

$$\Delta\left(\boldsymbol{\Theta}(t)\right) \triangleq \mathbb{E}\left[L\left(\boldsymbol{\Theta}(t+1)\right) - L\left(\boldsymbol{\Theta}(t)\right)|\boldsymbol{\Theta}(t)\right].$$

Using the Lyapunov optimization framework in [2] and [28], our strategy is to make transmission and packet dropping decisions to minimize a bound on the following "drift-plus-penalty" expression every slot:

$$\Delta\left(\boldsymbol{\Theta}(t)\right) - V\mathbb{E}\left[\hat{g}\left(\boldsymbol{\gamma}(t)\right)|\boldsymbol{\Theta}(t)\right]$$

where V is a nonnegative control parameter that is chosen as desired and will affect an explicit utility–delay tradeoff. Here, the "penalty" for slot t is considered to be -1 times the "reward" $\hat{g}(\boldsymbol{\gamma}(t))$. We later show that our resulting algorithm has a certain independence property, defined as follows.

Definition 1: We say that a control algorithm implemented over time has the *independence property* if for any slot t, every link l such that $H_l(t) > 0$ has a value of $T_l(t)$ that is independent of $\boldsymbol{\Theta}(t)$, $\mu_l(t)$, and $D_l(t)$.

To understand the above definition, suppose that $H_l(t) > 0$ for a certain slot t and queue l. Then, queue l has a packet in the head-of-line, and $T_l(t)$ is the random interarrival time between this head-of-line packet and the next packet. Because arrivals are independent over queues and i.i.d. over slots, the independence property arises naturally in algorithms that make control decisions up to time t that are independent of the $T_l(t)$ value. In this case, given that $H_l(t) > 0, T_l(t)$ is just a geometric random variable with success probability λ_l and mean $\mathbb{E}[T_l(t)|H_l(t) > 0] = 1/\lambda_l$.

Lemma 1: Every slot t, for any value of $\boldsymbol{\Theta}(t)$, and under any control policy that satisfies the independence property, the Lyapunov drift satisfies

$$\begin{aligned}\Delta\left(\boldsymbol{\Theta}(t)\right) \leq B &- \sum_l Z_l(t)\mathbb{E}\left[\lambda_l - D_l(t) - \gamma_l(t)|\boldsymbol{\Theta}(t)\right] \\ &- \sum_l \lambda_l H_l(t)\mathbb{E}\left[\left(\mu_l(t) + D_l(t)\right)T_l(t) - 1|\boldsymbol{\Theta}(t)\right]\end{aligned}$$

where B is a finite constant that does not depend on V.

Proof: The proof is given in Appendix B, where the constant B is also specified. ∎

Lemma 2: Every slot t, for any value of $\boldsymbol{\Theta}(t)$, and under any control policy that satisfies the independence property, we have

$$\begin{aligned}&\Delta\left(\boldsymbol{\Theta}(t)\right) - V\mathbb{E}\left[\hat{g}\left(\boldsymbol{\gamma}(t)\right)|\boldsymbol{\Theta}(t)\right] \\ &\leq B - V\mathbb{E}\left[\hat{g}\left(\boldsymbol{\gamma}(t)\right)|\boldsymbol{\Theta}(t)\right] \\ &\quad - \sum_l Z_l(t)\mathbb{E}\left[\lambda_l - D_l(t) - \gamma_l(t)|\boldsymbol{\Theta}(t)\right] \\ &\quad - \sum_l H_l(t)\mathbb{E}\left[\mu_l(t) + D_l(t) - \lambda_l|\boldsymbol{\Theta}(t)\right]\end{aligned}$$

where B is the same constant from Lemma 1.

Proof: Let $\chi(t)$ represent $[\boldsymbol{\Theta}(t); \mu_l(t) + D_l(t)]$. Using the law of iterated expectations, we have for any t and any queue l such that $H_l(t) > 0$

$$\begin{aligned}&\mathbb{E}\left[\left(\mu_l(t) + D_l(t)\right)T_l(t)|\boldsymbol{\Theta}(t)\right] \\ &= \mathbb{E}\left[\mathbb{E}\left[\left(\mu_l(t) + D_l(t)\right)T_l(t)|\chi(t)\right]|\boldsymbol{\Theta}(t)\right] \\ &= \mathbb{E}\left[\left(\mu_l(t) + D_l(t)\right)\mathbb{E}\left[T_l(t)|\chi(t)\right]|\boldsymbol{\Theta}(t)\right] \\ &= \frac{1}{\lambda_l}\mathbb{E}\left[\left(\mu_l(t) + D_l(t)\right)|\boldsymbol{\Theta}(t)\right]\end{aligned}$$

where we have used the fact that, by the independence property, if $H_l(t) > 0$, then $T_l(t)$ is independent of $\chi(t)$ and $\mathbb{E}[T_l(t)|\mathcal{X}(t)] = 1/\lambda_l$. Thus, for any slot t and any link l (regardless of whether or not $H_l(t) > 0$) we have[3]

$$\begin{aligned}&\lambda_l H_l(t)\mathbb{E}\left[\left(\mu_l(t) + D_l(t)\right)T_l(t) - 1|\boldsymbol{\Theta}(t)\right] \\ &\qquad = H_l(t)\mathbb{E}\left[\mu_l(t) + D_l(t) - \lambda_l|\boldsymbol{\Theta}(t)\right]. \quad (22)\end{aligned}$$

Lemma 2 follows by plugging this identity into Lemma 1 and subtracting $V\mathbb{E}[\hat{g}(\boldsymbol{\gamma}(t))|\boldsymbol{\Theta}(t)]$ from both sides. ∎

Our dynamic policy below is designed to make control decisions for $\boldsymbol{\gamma}(t)$, $\boldsymbol{D}(t)$, and $\boldsymbol{x}(t)$ (and hence $\boldsymbol{\mu}(t)$) to minimize the right-hand side of the drift-plus-penalty bound in Lemma 2. To analyze performance, we later show that this policy indeed satisfies the independence property.

F. Delay-Based Flow Control and Scheduling Algorithm

Every slot t, observe $\boldsymbol{Z}(t)$, $\boldsymbol{H}(t)$, and $\boldsymbol{S}(t)$, and perform the following operations, described as four control phases:

1) *Auxiliary Variable Selection*: Choose $\boldsymbol{\gamma}(t) = (\gamma_1(t), \ldots, \gamma_L(t))$ as the solution to the following:

$$\begin{aligned}\text{Maximize}: \quad & V\hat{g}\left(\boldsymbol{\gamma}(t)\right) - \sum_l Z_l(t)\gamma_l(t) \\ \text{Subject to}: \quad & -1 \leq \gamma_l(t) \leq 1 \qquad \text{for all } l \in \{1, \ldots, L\}.\end{aligned}$$

In the case of the separable utility function (4), this amounts to solving L single-variable concave optimizations over an interval and has a closed-form solution when $\hat{g}_l(\gamma_l)$ has a derivative with a closed-form inverse.

2) *Transmission Scheduling*: Observe $\boldsymbol{Z}(t), \boldsymbol{H}(t), \boldsymbol{S}(t)$ and choose a transmission vector $\boldsymbol{x}(t)$ to solve the following:

$$\begin{aligned}\text{Maximize}: \quad & \sum_l x_l(t)\min\left[H_l(t), Z_l(t)\right]\Psi_l\left(\boldsymbol{x}(t), \boldsymbol{S}(t)\right) \\ \text{Subject to}: \quad & \boldsymbol{x}(t) \in \mathcal{X}.\end{aligned}$$

3) *Packet Dropping*: For each link l that has a head-of-line packet that was not successfully transmitted in the scheduling phase (either because its transmission was not attempted or its transmission failed), drop the packet if $Z_l(t) \leq H_l(t)$. Else, keep it in the head-of-line.

4) *Queue Updates*: Update the virtual queues $Z_l(t)$ according to (19), using the values of $\gamma_l(t)$ and $D_l(t)$ as determined from the above auxiliary variable and packet dropping phases. Also update the actual queues and the head-of-line values according to (1) and (21) by simply removing any packet that was either successfully transmitted or dropped.

[3]Note that the identity (22) is trivially true for the case $H_l(t) = 0$.

Now, for each link l, define the quantity $H_{l,\max}$ as follows:

$$H_{l,\max} \triangleq \lceil V\nu_l \rceil + 2. \tag{23}$$

Note that $H_{l,\max}$ is $O(V)$.

Theorem 1 (Algorithm Performance): Suppose all queues are initially empty and that Assumption 1 holds. If the above control policy is implemented with a particular constant $V > 0$, then achieved utility satisfies

$$\liminf_{t\to\infty} g\left(\overline{\boldsymbol{y}}(t)\right) \geq g^* - B/V \tag{24}$$

where $\overline{\boldsymbol{y}}(t)$ is defined in (8), B is the constant from Lemma 1, and g^* is the optimal utility, defined as the supremum utility for problem (9)–(11). Finally, for all links l, we have

$$Q_l(t) \leq H_l(t) \leq H_{l,\max} \qquad \forall t \in \{0,1,2,\ldots\} \tag{25}$$

$$Z_l(t) \leq H_{l,\max} \qquad \forall t \in \{0,1,2,\ldots\}. \tag{26}$$

The above theorem provides the strong *deterministic guarantee* that head-of-line packets in queue l always have delay less than or equal to $H_{l,\max}$, and hence all nondropped packets in queue l have delay upper bounded by this value. Thus, if we wish to enforce the constraint that worst-case delay in all queues is less than or equal to a given constant $D_{\max}$, we can choose V to satisfy $\lceil V \max_l\{\nu_l\}\rceil + 2 = D_{\max}$. As the delay constraint $D_{\max}$ is relaxed, the value V goes to infinity, and hence by (24) we know that utility converges to the optimal value g^*. Specifically, for a given delay guarantee $D_{\max}$, the achieved utility is guaranteed to be within $O(1/D_{\max})$ of the optimal utility g^*.

Remark 1: In the case when link reliability is also affected by a set of additional decision parameters $I(t) \in \mathcal{I}$, as discussed in Section II-A, the transmission scheduling decision in phase 2 of the algorithm is modified to maximize

$$\sum_l x_l(t) \min\left[H_l(t), Z_l(t)\right] \Psi_l\left(\boldsymbol{x}(t), \boldsymbol{S}(t), I(t)\right) \tag{27}$$

subject to $\boldsymbol{x}(t) \in \mathcal{X}$ and $I(t) \in \mathcal{I}$. Theorem 1 holds exactly as stated under this modification, with the understanding that the optimal utility value g^* may change due to the increased options for scheduling.

Remark 2: The deterministic guarantees (25) and (26) on queue size and delay are sample path results that hold always, regardless of whether or not the state vector $\boldsymbol{S}(t)$ is i.i.d. over slots and/or the arrival vector $\boldsymbol{A}(t)$ has independent entries and is i.i.d. over slots. The independence assumptions are used only to prove the utility guarantee (24).

IV. Performance Analysis

Here, we prove Theorem 1. We first prove the deterministic bounds (25) and (26), which use a preliminary lemma.

Lemma 3: If $Z_l(t) > V\nu_l$ for a particular slot t and link l, then the auxiliary variable selection in the first phase of the control algorithm chooses $\gamma_l(t) = -1$ for that slot.

Proof: The value of $\gamma_l(t)$ is determined by maximizing $V\hat{g}(\boldsymbol{\gamma}) - \sum_m Z_m(t)\gamma_m$ over $-\mathbf{1} \leq \boldsymbol{\gamma} \leq \mathbf{1}$. By (13), we know that for any vector $\boldsymbol{\gamma}$ such that $-\mathbf{1} \leq \boldsymbol{\gamma} \leq \mathbf{1}$

$$V\hat{g}(\boldsymbol{\gamma}) - \sum_m Z_m(t)\gamma_m \leq V\hat{g}\left(\boldsymbol{\gamma}_l'\right) + V\nu_l(\gamma_l+1) - \sum_m Z_m(t)\gamma_m$$

where $\boldsymbol{\gamma}_l'$ is formed from $\boldsymbol{\gamma}$ by replacing entry γ_l with -1. Because $V\nu_l < Z_l(t)$, the right-hand side of the above bound is maximized at $\gamma_l = -1$, so that

$$V\hat{g}(\boldsymbol{\gamma}) - \sum_m Z_m(t)\gamma_m \leq V\hat{g}\left(\boldsymbol{\gamma}_l'\right) - \sum_{m\neq l} Z_m(t)\gamma_m + Z_l(t)$$

and equality holds if and only if $\gamma_l = -1$. Hence, the auxiliary variable optimization *must* choose $\gamma_l(t) = -1$. ■

Proof [Deterministic Bounds (25) and (26)]: Fix a link l. We first show that $Z_l(t) \leq \lceil V\nu_l \rceil + 2$ for all $t \geq 0$. This clearly holds for $t = 0$ when all queues are empty. Suppose it holds at some time t. We prove it also holds for time $t+1$.

Note from (19) that $Z_l(t)$ can increase by at most 2 every slot (since $D_l(t) \leq 1$ and $\gamma_l(t) \leq 1$ for all t). If $Z_l(t) \leq \lceil V\nu_l \rceil$, then $Z_l(t+1) \leq \lceil V\nu_l \rceil + 2$ and the bound holds. Else, we have $Z_l(t) > \lceil V\nu_l \rceil$, and so by Lemma 3 we know that $\gamma_l(t) = -1$. Because $D_l(t) \leq 1$ for all t, we have

$$D_l(t) + \gamma_l(t) \leq 0. \tag{28}$$

Hence, from the update equation for $Z_l(t+1)$ in (19), we have $Z_l(t+1) \leq Z_l(t) \leq \lceil V\nu_l \rceil + 2$, so the bound again holds. Hence, $Z_l(t) \leq \lceil V\nu_l \rceil + 2$ for all t.

Similarly, we use induction to show that $H_l(t) \leq \lceil V\nu_l \rceil + 2$ for all t. It clearly holds for $t = 0$. Assume it holds for a general slot t. First suppose that $H_l(t) \leq \lceil V\nu_l \rceil + 1$. Because the head-of-line delay can increase by at most 1 every slot, we know that $H_l(t+1) \leq \lceil V\nu_l \rceil + 2$, and we are done. In the opposite case when $H_l(t) > \lceil V\nu_l \rceil + 1$, we know that $H_l(t) = \lceil V\nu_l \rceil + 2$ (since, by assumption for slot t, $H_l(t)$ must be an integer that is bounded by $\lceil V\nu_l \rceil + 2$). It follows that $H_l(t) \geq Z_l(t)$, and so by the packet-dropping procedure in phase 3 of the algorithm, the head-of-line packet will either be successfully transmitted on this slot or dropped. It follows that in slot $t+1$ there will be either *no* head-of-line packet (so that $H_l(t+1) = 0$), or there will be a *new* head-of-line packet, in which case its delay is no more than the delay of the previous head-of-line packet, so that $H_l(t+1) \leq H_l(t) \leq \lceil V\nu_l \rceil + 2$.

Finally, because there is at most one packet arrival to link l per slot, it is clear that the number of packets in the queue is no more than the current delay of the head-of-line packet, so that $Q_l(t) \leq H_l(t)$ for all t. ■

Lemma 4: The control policy chooses decision variables that minimize the right-hand side of the drift-plus-penalty inequality in Lemma 2.

Proof: See Appendix A. ■

Lemma 5: Assuming arrivals $A_l(t)$ are independent over links and i.i.d. over slots, the control policy has the independence property needed for Lemma 2 to hold.

Proof: This follows immediately by noting that for any slot t and any link l such that $H_l(t) > 0$, the value of $T_l(t)$, being the interarrival time between the head-of-line packet at link l and the next packet, has not affected any queue values or decisions up to (and including) slot t. ■

We now prove the utility bound (24). We first use a preliminary lemma, which demonstrates that our structure of making a transmission decision *first*, and then choosing to drop at most one packet per queue, does not limit optimality.

Lemma 6: Let $\boldsymbol{y}^*$ be the optimal time-average throughput vector that solves (9)–(11), so that $g(\boldsymbol{y}^*) = g^*$, and $\boldsymbol{y}^*$ satisfies the constraints $\boldsymbol{y}^* \in \Lambda$ and $\boldsymbol{0} \leq \boldsymbol{y}^* \leq \boldsymbol{\lambda}$. Then, there is an $\boldsymbol{S}$-only algorithm that is independent of the current $\boldsymbol{\Theta}(t)$ and that observes $\boldsymbol{S}(t)$ and makes a randomized transmission decision $\boldsymbol{x}^*(t)$ (leading to a vector $\boldsymbol{\mu}^*(t)$ of successful transmissions), and then makes randomized packet drop decisions $\boldsymbol{D}^*(t)$ in reaction to the ACK/NACK feedback, such that $\mu_l^*(t) + D_l^*(t) \leq 1$ for all l, and

$$\mathbb{E}\left[\boldsymbol{\mu}^*(t)\right] = \boldsymbol{y}^* \quad \mathbb{E}\left[\boldsymbol{D}^*(t)\right] = \boldsymbol{\lambda} - \boldsymbol{y}^*. \tag{29}$$

Proof: The proof follows easily from the fact (3) and is omitted for brevity. ∎

Proof [Utility Bound (24)]: Because our control algorithm satisfies the independence property, the drift inequality in Lemma 2 holds. Furthermore (by Lemma 4), our policy minimizes the right-hand side of this inequality every slot. We thus know

$$\begin{aligned}
&\Delta\left(\boldsymbol{\Theta}(t)\right) - V\mathbb{E}\left[\hat{g}\left(\boldsymbol{\gamma}(t)\right)|\boldsymbol{\Theta}(t)\right] \\
&\quad \leq B - V\mathbb{E}\left[\hat{g}\left(\boldsymbol{\gamma}^*(t)\right)|\boldsymbol{\Theta}(t)\right] \\
&\qquad - \sum_l Z_l(t)\mathbb{E}\left[\lambda_l - D_l^*(t) - \gamma_l^*(t)|\boldsymbol{\Theta}(t)\right] \\
&\qquad - \sum_l H_l(t)\mathbb{E}\left[\mu_l^*(t) + D_l^*(t) - \lambda_l|\boldsymbol{\Theta}(t)\right]
\end{aligned}$$

where $\boldsymbol{\gamma}^*(t)$, $\boldsymbol{D}^*(t)$, and $\boldsymbol{\mu}^*(t)$ are from any $\boldsymbol{S}$-only policy (independent of $\boldsymbol{\Theta}(t)$). Taking expectations of both sides of the above and using the law of iterated expectations yields

$$\begin{aligned}
&\mathbb{E}\left[L\left(\boldsymbol{\Theta}(t+1)\right)\right] - \mathbb{E}\left[L\left(\boldsymbol{\Theta}(t)\right)\right] - V\mathbb{E}\left[\hat{g}\left(\boldsymbol{\gamma}(t)\right)\right] \\
&\quad \leq B - V\mathbb{E}\left[\hat{g}\left(\boldsymbol{\gamma}^*(t)\right)\right] - \sum_l \mathbb{E}\left[Z_l(t)\right]\mathbb{E}\left[\lambda_l - D_l^*(t) - \gamma_l^*(t)\right] \\
&\qquad - \sum_l \mathbb{E}\left[H_l(t)\right]\mathbb{E}\left[\mu_l^*(t) + D_l^*(t) - \lambda_l\right].
\end{aligned} \tag{30}$$

Now choose the alternative auxiliary variable decision

$$\boldsymbol{\gamma}^*(t) = \boldsymbol{y}^*. \tag{31}$$

This is a feasible decision because the optimal vector $\boldsymbol{y}^*$ satisfies $0 \leq y_l^* \leq 1$ for all l, and so clearly $-1 \leq \gamma_l^* \leq 1$ for all l. Furthermore, choose $\boldsymbol{D}^*(t)$ and $\boldsymbol{\mu}^*(t)$ as the $\boldsymbol{S}$-only decisions that yield (29) from Lemma 6. Plugging (31) and (29) into the right-hand side of (30) and using the fact that $g(\boldsymbol{y}^*) = g^*$ yields

$$\mathbb{E}\left[L\left(\boldsymbol{\Theta}(t+1)\right)\right] - \mathbb{E}\left[L\left(\boldsymbol{\Theta}(t)\right)\right] - V\mathbb{E}\left[\hat{g}\left(\boldsymbol{\gamma}(t)\right)\right] \leq B - Vg^*.$$

The above holds for all $t \geq 0$. Summing over $\tau \in \{0, \ldots, t-1\}$ and dividing by t yields

$$\frac{\mathbb{E}\left[L\left(\boldsymbol{\Theta}(t)\right)\right] - \mathbb{E}\left[L\left(\boldsymbol{\Theta}(0)\right)\right]}{t} - \frac{V}{t}\sum_{\tau=0}^{t-1}\mathbb{E}\left[\hat{g}\left(\boldsymbol{\gamma}(\tau)\right)\right] \leq B - Vg^*.$$

Using the fact that $L(\cdot) \geq 0$ and rearranging terms yields

$$\frac{1}{t}\sum_{\tau=0}^{t-1}\mathbb{E}\left[\hat{g}\left(\boldsymbol{\gamma}(\tau)\right)\right] \geq g^* - B/V - \frac{\mathbb{E}\left[L\left(\boldsymbol{\Theta}(0)\right)\right]}{Vt}.$$

Using Jensen's inequality in the concave function $\hat{g}(\cdot)$ yields

$$\hat{g}\left(\overline{\boldsymbol{\gamma}}(t)\right) \geq g^* - B/V - \frac{\mathbb{E}\left[L\left(\boldsymbol{\Theta}(0)\right)\right]}{Vt} \tag{32}$$

where $\overline{\boldsymbol{\gamma}}(t)$ is defined as

$$\overline{\boldsymbol{\gamma}}(t) \triangleq \frac{1}{t}\sum_{\tau=0}^{t-1}\mathbb{E}\left[\boldsymbol{\gamma}(\tau)\right].$$

However, because $Z_l(t) \leq H_{l,\max}$ for all l and all t, we have from (20)

$$\overline{\boldsymbol{y}}(t) + \boldsymbol{H}_{\max}/t \geq \overline{\boldsymbol{\gamma}}(t)$$

where $\boldsymbol{H}_{\max} = (H_{l,\max})|_{l=1}^L$. For all t, we have $-\boldsymbol{1} \leq \overline{\boldsymbol{\gamma}}(t) \leq \boldsymbol{1}$ and $\boldsymbol{0} \leq \overline{\boldsymbol{y}}(t) \leq \boldsymbol{1}$ [where the latter can be shown by definition of $\overline{\boldsymbol{y}}(t)$ in (8)]. Thus

$$[\overline{\boldsymbol{y}}(t) + \boldsymbol{H}_{\max}/t]_0^1 \geq \overline{\boldsymbol{\gamma}}(t)$$

where $[x]_0^1$ is equal to x if $0 \leq x \leq 1$, 1 if $x > 1$, and 0 else. Plugging this into (32) and using the fact that $\hat{g}(\cdot)$ is nondecreasing in each entry yields

$$\hat{g}\left([\overline{\boldsymbol{y}}(t) + \boldsymbol{H}_{\max}/t]_0^1\right) \geq g^* - B/V - \frac{\mathbb{E}\left[L\left(\boldsymbol{\Theta}(0)\right)\right]}{Vt}. \tag{33}$$

The above holds for all t. By continuity of $\hat{g}(\cdot)$ and the facts that $\boldsymbol{0} \leq \overline{\boldsymbol{y}}(t) \leq \boldsymbol{1}$ and $H_{l,\max}/t \to 0$, we have

$$\liminf_{t\to\infty} \hat{g}\left(\overline{\boldsymbol{y}}(t)\right) \geq g^* - B/V. \tag{34}$$

Because $g(\boldsymbol{y}) = \hat{g}(\boldsymbol{y})$ whenever $\boldsymbol{0} \leq \boldsymbol{y} \leq \boldsymbol{1}$, we have

$$\liminf_{t\to\infty} \hat{g}\left(\overline{\boldsymbol{y}}(t)\right) = \liminf_{t\to\infty} g\left(\overline{\boldsymbol{y}}(t)\right).$$

Using this in (34) proves (24). ∎

V. Unknown Arrival Rates λ_l

Consider now a modification of the algorithm that performs all operations in the same way, with the exception that the virtual queue update rule (19) for each $l \in \{1, \ldots, L\}$ is replaced by the following:

$$Z_l(t+1) = \max\left[Z_l(t) - A_l(t-W) + D_l(t) + \gamma_l(t), 0\right] \tag{35}$$

where the time-shift constant W is set to the following value:

$$W \triangleq \max_{l\in\{1,\ldots,L\}} \lceil V\nu_l \rceil + 2. \tag{36}$$

Unlike the previous algorithm, this modified algorithm can be implemented without knowledge of the λ_l values. Here, we show that it also ensures all queue sizes are deterministically $O(V)$, with $O(1/V)$ deviation from the optimal utility g^*.

Note that the deterministic bounds (25) and (26) still hold, so that $Q_l(t) \leq H_l(t) \leq \lceil V\nu_l \rceil + 2$ and $Z_l(t) \leq \lceil V\nu_l \rceil + 2$ for all t and all l. In particular, all packets depart the system within W slots. This is because the proof of these bounds, given in Section IV, only uses the fact that $Z_l(t)$ can increase by at most $D_l(t) + \gamma_l(t)$ every slot t under the old update rule (19), which

is still true under the modified update rule (35). Thus, it remains only to show that the modified algorithm comes within $O(1/V)$ of the optimal utility g^*.

A. Modified Drift Bounds

We have the following lemmas, which are modifications of Lemmas 1 and 2.

Lemma 7: Under the given modified algorithm, the Lyapunov drift satisfies the following for all slots t and all possible $\mathbf{\Theta}(t)$

$$\Delta(\mathbf{\Theta}(t)) \leq C - \sum_{l=1}^{L} Z_l(t)\mathbb{E}\left[A_l(t-W) - D_l(t) - \gamma_l(t)|\mathbf{\Theta}(t)\right] - \sum_{l=1}^{L} \lambda_l H_l(t)\mathbb{E}\left[(\mu_l(t) + D_l(t))\, T_l(t) - 1|\mathbf{\Theta}(t)\right]$$

where C is a finite constant that does not depend on V.

Proof: The proof is similar to that of Lemma 1 and is omitted for brevity. ∎

Lemma 8: Under the given modified algorithm, the Lyapunov drift satisfies the following for all slots t and all possible $\mathbf{\Theta}(t)$:

$$\begin{aligned}&\Delta(\mathbf{\Theta}(t)) - V\mathbb{E}\left[\hat{g}\left(\boldsymbol{\gamma}(t)\right)|\mathbf{\Theta}(t)\right]\\ &\quad\leq C - V\mathbb{E}\left[\hat{g}\left(\boldsymbol{\gamma}(t)\right)|\mathbf{\Theta}(t)\right]\\ &\qquad - \sum_{l=1}^{L} Z_l(t)\mathbb{E}\left[A_l(t-W) - D_l(t) - \gamma_l(t)|\mathbf{\Theta}(t)\right]\\ &\qquad - \sum_{l=1}^{L} H_l(t)\mathbb{E}\left[(\mu_l(t) + D_l(t)) - \lambda_l|\mathbf{\Theta}(t)\right]\end{aligned}$$

where C is the same constant from Lemma 7.

Proof: We show that the last term on the right-hand side of the bound in Lemma 7 is the same as the last term on the right-hand side of the bound in this lemma. To see this, note that for all t, all $\mathbf{\Theta}(t)$, and each l, the following trivially holds whenever $H_l(t) = 0$ (because $0 = 0$):

$$\begin{aligned}&\lambda_l H_l(t)\mathbb{E}\left[(\mu_l(t) + D_l(t))\, T_l(t) - 1|\mathbf{\Theta}(t)\right]\\ &\qquad = H_l(t)\mathbb{E}\left[(\mu_l(t) + D_l(t)) - \lambda_l|\mathbf{\Theta}(t)\right]. \quad (37)\end{aligned}$$

We now show that (37) also holds when $H_l(t) > 0$. In this case, there is a packet at the head-of-line in queue l, and $T_l(t)$ is the interarrival time between this packet and the next. This interarrival time has not affected any of the control decisions or queue states up to and including slot t because the virtual queue (35) only uses arrival information of packets that arrived W or more slots ago, and all these packets have already departed. Thus, given $H_l(t) > 0$, the value $T_l(t)$ is independent of $\mathbf{\Theta}(t)$, $\mu_l(t)$, and $D_l(t)$, so that

$$\begin{aligned}&\mathbb{E}\left[(\mu_l(t) + D_l(t))\, T_l(t)|\mathbf{\Theta}(t)\right]\\ &\quad = \mathbb{E}\left[T_l(t)|\mathbf{\Theta}(t)\right]\mathbb{E}\left[\mu_l(t) + D_l(t)|\mathbf{\Theta}(t)\right]\\ &\quad = (1/\lambda_l)\mathbb{E}\left[\mu_l(t) + D_l(t)|\mathbf{\Theta}(t)\right]\end{aligned}$$

from which we obtain (37). ∎

It is easy to see that the algorithm that minimizes the right-hand side of the inequality in Lemma 8 is the same as that which minimizes the right-hand side of the inequality in Lemma 2, and so our algorithm accomplishes this. Thus, for all t and all $\mathbf{\Theta}(t)$, we have

$$\begin{aligned}&\Delta(\mathbf{\Theta}(t)) - V\mathbb{E}\left[\hat{g}\left(\boldsymbol{\gamma}(t)\right)|\mathbf{\Theta}(t)\right]\\ &\quad\leq C - V\mathbb{E}\left[\hat{g}\left(\boldsymbol{\gamma}^*(t)\right)|\mathbf{\Theta}(t)\right]\\ &\qquad - \sum_{l=1}^{L} Z_l(t)\mathbb{E}\left[A_l(t-W) - D_l^*(t) - \gamma_l^*(t)|\mathbf{\Theta}(t)\right]\\ &\qquad - \sum_{l=1}^{L} H_l(t)\mathbb{E}\left[(\mu_l^*(t) + D_l^*(t)) - \lambda_l|\mathbf{\Theta}(t)\right] \quad (38)\end{aligned}$$

where $\boldsymbol{\gamma}^*(t), D_l^*(t), \mu_l^*(t)$ are the result of any alternative decisions for slot t. However, we have

$$\begin{aligned}&Z_l(t)\left[A_l(t-W) - D_l^*(t) - \gamma_l^*(t)\right]\\ &\quad = Z_l(t-W)\left[A_l(t-W) - D_l^*(t) - \gamma_l^*(t)\right]\\ &\qquad + \left(Z_l(t) - Z_l(t-W)\right)\left[A_l(t-W) - D_l^*(t) - \gamma_l^*(t)\right]\\ &\quad \leq Z_l(t-W)\left[A_l(t-W) - D_l^*(t) - \gamma_l^*(t)\right] + 4W\end{aligned}$$

which follows because $Z_l(\tau)$ can change by at most 2 on any slot τ, and

$$|A_l(t-W) - D_l^*(t) - \gamma_l^*(t)| \leq 2.$$

Plugging this into the right-hand side of (38) yields

$$\begin{aligned}&\Delta(\mathbf{\Theta}(t)) - V\mathbb{E}\left[\hat{g}\left(\boldsymbol{\gamma}(t)\right)|\mathbf{\Theta}(t)\right]\\ &\quad\leq \tilde{C} - V\mathbb{E}\left[\hat{g}\left(\boldsymbol{\gamma}^*(t)\right)|\mathbf{\Theta}(t)\right]\\ &\qquad - \sum_{l=1}^{L} Z_l(t-W)\mathbb{E}\left[A_l(t-W) - D_l^*(t) - \gamma_l^*(t)|\mathbf{\Theta}(t)\right]\\ &\qquad - \sum_{l=1}^{L} H_l(t)\mathbb{E}\left[(\mu_l^*(t) + D_l^*(t)) - \lambda_l|\mathbf{\Theta}(t)\right]\end{aligned}$$

where we define

$$\tilde{C} \triangleq C + 4W.$$

Taking expectations of both sides and using iterated expectations yields

$$\begin{aligned}&\mathbb{E}\left[L\left(\mathbf{\Theta}(t+1)\right)\right] - \mathbb{E}\left[L\left(\mathbf{\Theta}(t)\right)\right] - V\mathbb{E}\left[\hat{g}\left(\boldsymbol{\gamma}(t)\right)\right]\\ &\quad\leq \tilde{C} - V\mathbb{E}\left[\hat{g}\left(\boldsymbol{\gamma}^*(t)\right)\right]\\ &\qquad - \sum_{l=1}^{L}\mathbb{E}\left[Z_l(t-W)\left[\lambda_l - D_l^*(t) - \gamma_l^*(t)\right]\right]\\ &\qquad - \sum_{l=1}^{L}\mathbb{E}\left[H_l(t)\left[\mu_l^*(t) + D_l^*(t) - \lambda_l\right]\right] \quad (39)\end{aligned}$$

where we have used the fact that

$$\mathbb{E}\left[Z_l(t-W)A_l(t-W)\right] = \mathbb{E}\left[Z_l(t-W)\right]\lambda_l$$

which follows because arrivals are i.i.d. over slots, so that $A_l(t-W)$ is independent of $Z_l(t-W)$.

Rates (λ_{ij})

.45	.1	.4
.1	.7	.15
.4	.15	.4

Achieved Throughput $(\overline{y}_{ij})$

.4497	.0998	.3996
.1002	.7000	.1499
.4001	.1504	.4005

Avg. Delay

10.5	10.6	11.1
10.9	9.11	11.0
11.6	11.3	11.5

Worst Delay

65	69	63
71	54	66
70	75	69

Fig. 2. Simulation for 3 × 3 switch with feasible traffic ($V = 100$).

We now plug the values $\boldsymbol{\gamma}^*(t), D_l^*(t), \mu_l^*(t)$ from the $\boldsymbol{S}$-only policy that we used for the proof of the original algorithm, which makes decisions independent of $\boldsymbol{\Theta}(t)$ to yield

$$\begin{aligned}\boldsymbol{\gamma}^*(t) &= \boldsymbol{y}^* \\ \hat{g}(\boldsymbol{y}^*) &= g^* \\ \mathbb{E}\left[\lambda_l - D_l^*(t) - \gamma_l^*(t)|\boldsymbol{\Theta}(t)\right] &= 0 \\ \mathbb{E}\left[\mu_l^*(t) + D_l^*(t) - \lambda_l|\boldsymbol{\Theta}(t)\right] &= 0\end{aligned}$$

The bound (39) becomes

$$\mathbb{E}\left[L\left(\boldsymbol{\Theta}(t+1)\right)\right] - \mathbb{E}\left[L\left(\boldsymbol{\Theta}(t)\right)\right] - V\mathbb{E}\left[\hat{g}\left(\boldsymbol{\gamma}(t)\right)\right] \leq \tilde{C} - Vg^*$$

from which we obtain the following performance bound in the same way as before [see (32)–(34)]:

$$\liminf_{t\to\infty} g\left(\overline{\boldsymbol{y}}(t)\right) \geq g^* - \tilde{C}/V.$$

VI. Simulations

This section presents simulation results for a 3 × 3 packet switch and a two-user wireless downlink with time-varying channels.

A. Scheduling for a 3 × 3 Packet Switch

Here, we consider a crossbar constrained 3 × 3 packet switch, having three input ports and three output ports (as in [5] and [7]). There are nine queues $Q_{ij}(t)$, representing packets that arrived to input port i that must be delivered to output port j, for $i \in \{1,2,3\}$, $j \in \{1,2,3\}$. Scheduling matrices are chosen every slot within the set of six permutation matrices, so that at most one packet is served per input and per output on a given slot. Arrival processes to each queue are independent Bernoulli processes, i.i.d. over slots with rates (λ_{ij}). We simulate the modified delay-based utility maximization algorithm of Section V, which does not require knowledge of the arrival rates (λ_{ij}). All simulations are over 1 million slots. The utility function of achieved throughput $\boldsymbol{y} = (y_{ij})$ is

$$g(\boldsymbol{y}) = \sum_{i=1}^{3}\sum_{j=1}^{3}\log(1+y_{ij})$$

where $\log(\cdot)$ denotes the natural logarithm. We choose V as a positive integer, so that the algorithm guarantees a worst-case delay of $V+2$ slots.

We first consider a switch with *feasible* input rates (λ_{ij}), given in the first panel of Fig. 2. The rates are chosen so that all input ports and output ports have a loading of 0.95. For example, the loading of input port 1 is $0.45 + 0.1 + 0.4$, being the sum of the rates in the first row of the (λ_{ij}) matrix. Because input rates are inside the capacity region of the switch, the utility optimal throughput matrix is $(y_{ij}^*) = (\lambda_{ij})$. Thus, the algorithm should learn to drop as few packets as possible. We use $V = 100$, which guarantees a worst-case delay of $V+2 = 102$ slots. The resulting throughput matrix and average delays are shown in Fig. 2. The figure shows that average delays are less than 12 slots, while the achieved throughput is almost the same as the arrival rates (so the algorithm indeed learns to drop almost no packets).[4] The worst-case delays are much less than the guarantee of 102 slots. While not shown in the figure, we note that a simulation for the case $V = 50$ yields almost the same throughput, does not significantly change average delay, but reduces the largest observed delay from 75 to 47 slots.

We next test the case when input rates (λ_{ij}) exceed the switch capacity region, so that the system is in overload and must drop packets. We use rates (λ_{ij}) given by

$$(\lambda_{ij}) = \begin{array}{|c|c|c|}\hline .9 & .2 & .3 \\ \hline 0 & .4 & .2 \\ \hline 0 & .5 & 0 \\ \hline\end{array}. \qquad (40)$$

The sum of the arrival rates to the first input port is 1.4, which is larger than 1. Fig. 3(a) shows results for the case $V = 100$. The achieved throughput $(\overline{y}_{ij})$ is within three significant digits of the utility-optimal throughput matrix (y_{ij}^*). Average delays and worst-case delays are also shown. Fig. 3(b) and 3(c) shows results for $V = 50$ and $V = 25$. It is seen that delays reduce when V is decreased, with a consequent deviation from the ideal throughput.

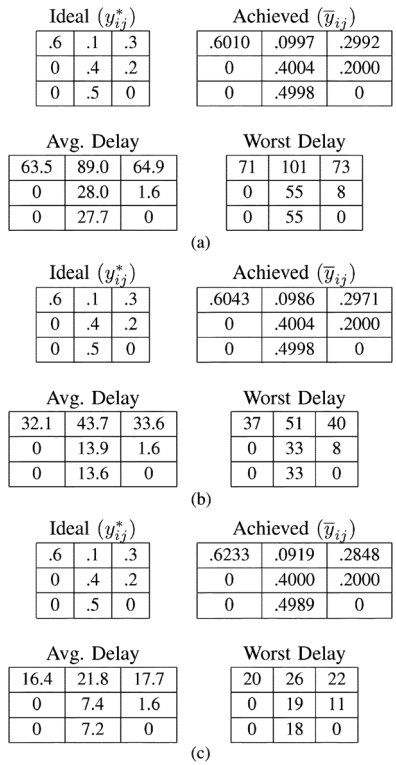


Fig. 3. Simulation of a 3 × 3 switch with overloaded traffic. The arrival rates (λ_{ij}) are given in (40). Each $A_{ij}(t)$ process is i.i.d. over slots. (a) Performance for overloaded switch ($V = 100$). (b) Performance for overloaded switch ($V = 50$). (c) Performance for overloaded switch ($V = 25$).

[4]For this case of feasible traffic with $V = 100$, the algorithm dropped only 11 packets during the course of the 1-million-slot simulation.

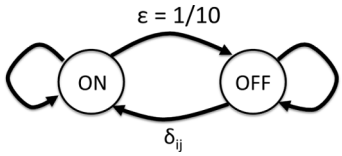


Fig. 4. Two-state ON/OFF Markov chain $M_{ij}(t)$.

Ideal (y^*_{ij})

.6	.1	.3
0	.4	.2
0	.5	0

Achieved $(\overline{y}_{ij})$

.6048	.0954	.2998
0	.4001	.1985
0	.5025	0

Avg. Delay

251.7	343.0	257.8
0	115.1	9.9
0	111.5	0

Worst Delay

294	401	317
0	280	78
0	280	0

(a)

Ideal (y^*_{ij})

.6	.1	.3
0	.4	.2
0	.5	0

Achieved $(\overline{y}_{ij})$

.6187	.0893	.2919
0	.3989	.1985
0	.5009	0

Avg. Delay

125.9	167.9	131.1
0	61.3	9.52
0	57.7	0

Worst Delay

152	201	169
0	150	81
0	140	0

(b)

Ideal (y^*_{ij})

.6	.1	.3
0	.4	.2
0	.5	0

Achieved $(\overline{y}_{ij})$

.6400	.0824	.2777
0	.3929	0.1985
0	.4953	0

Avg. Delay

62.9	83.4	67.3
0	33.9	8.75
0	30.6	0

Worst Delay

81	101	90
0	83	70
0	74	0

(c)

Fig. 5. Simulation of a 3 × 3 switch with Markov ON/OFF arrival processes. The arrival rates (λ_{ij}) are given in (40). The worst-case delay bound is $V + 2$. (a) Performance for non-i.i.d. switch $(V = 400)$. (b) Performance for non-i.i.d. switch $(V = 200)$. (c) Performance for non-i.i.d. switch $(V = 100)$.

B. Packet Switch With Bursty Arrivals

Here, we consider the same switch and the same algorithm as in Section VI-A, but we change the arrivals to non-i.i.d. "bursty" processes. Specifically, for each $i, j \in \{1, 2, 3\}$, let $M_{ij}(t)$ be an independent 2-state ON/OFF Markov chain with transition probabilities ϵ and δ_{ij}, as shown in Fig. 4. We have $A_{ij}(t) = 0$ if $M_{ij}(t) = \text{OFF}$, and $A_{ij}(t) = 1$ if $M_{ij}(t) = \text{ON}$. We use $\epsilon = 1/10$, which means that ON periods have average size $1/\epsilon = 10$ slots. The OFF periods have average size $1/\delta_{ij}$, and δ_{ij} is set to ensure a desired arrival rate λ_{ij}. Specifically, we have $\lambda_{ij} = \delta_{ij}/(\delta_{ij} + \epsilon)$, and so $\delta_{ij} = \epsilon\lambda_{ij}/(1 - \lambda_{ij})$. We use the same arrival rates (λ_{ij}) as given in (40), for which the above δ_{ij} values are valid probabilities when $\epsilon = 1/10$.

We expect the admitted rates to approach the same ideal values given in Fig. 3 for the i.i.d. case. This is indeed what happens. However, we require larger values of V to achieve the same utility performance for this non-i.i.d. scenario, which leads to larger delay. This is intuitive: These bursty arrivals create more congestion and delay than i.i.d. arrivals. The resulting throughput values for $V = 400, 200, 100$ are shown in Fig. 5. The case $V = 400$ in Fig. 5(a) yields rates very close to the ideal. The case $V = 100$ in Fig. 5(c) has rates that deviate significantly from this ideal, in contrast to the $V = 100$ case for the i.i.d. system in Fig. 3(a).

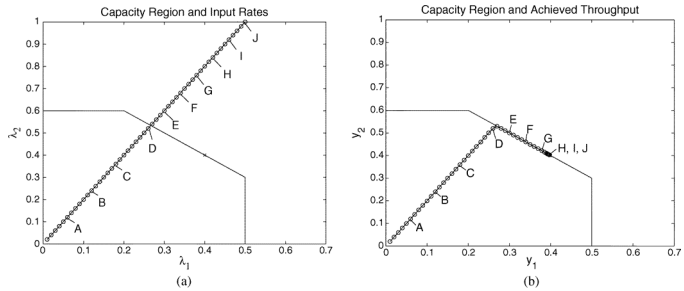


Fig. 6. Capacity region of the two-user wireless downlink. (a) Input rate vectors (λ_1, λ_2) simulated, with sample points $A, B, \ldots, H, I, J$. (b) Resulting achieved throughput vectors $(\overline{y}_1, \overline{y}_2)$.

C. Opportunistic Scheduling for a Two-User Wireless Downlink

Here, we consider a two-user wireless downlink with ON/OFF channels. The channel state processes $S_i(t)$ are independent and i.i.d. over slots with $\Pr[S_1(t) = \text{ON}] = 0.5$ and $\Pr[S_2(t) = \text{ON}] = 0.6$. Arrivals $A_1(t)$ and $A_2(t)$ are independent Bernoulli processes, i.i.d. over slots with rates λ_1 and λ_2. Every slot, the network controller observes the channel states $(S_1(t), S_2(t))$ and chooses a single queue to serve, transmitting exactly one packet over a served channel that is ON, and no packets over a channel that is not served or that is OFF. The capacity region is shown in Fig. 6.

We simulate the delay-based algorithm of Section III-F, which uses knowledge of the arrival rates (λ_1, λ_2). We use a utility function

$$g(y_1, y_2) = \log(1 + y_1) + \log(1 + y_2)$$

and use $V = 1000$, which yields near-optimal utility. All simulations run for 4 million slots. We create 50 different simulation runs, for arrival rates (λ_1, λ_2) that scale linearly toward the point (0.5, 1.0), as shown in the left panel of Fig. 6. The resulting achieved throughput vectors $(\overline{y}_1, \overline{y}_2)$ are shown in the right panel of Fig. 6.

The example arrival rate points A, B, C, D in the left panel of Fig. 6 are all inside the capacity region, and hence the achieved throughputs should be the same. This is indeed the case, as shown by the corresponding example points A, B, C, D on the right panel. Arrival point E in the left panel is outside of the capacity region, and its optimal achieved throughput is shown on the boundary point E in the right panel. Note that once the arrival rates exceed the point H in the left panel, the achieved throughput is the same and is very close to (0.4, 0.4) (shown as H, I, J in the right panel). While these achieved throughputs are for the delay-based algorithm with known arrival rates, we note that we also simulated the modified delay-based algorithm with unknown arrival rates, as well as the queue-based algorithm of [2] (version CLC2 in [2]). The achieved throughputs for all of these algorithms are nearly identical, and the picture in the right panel of Fig. 6 would look the same for all three algorithms.

We next consider the throughput and delay as a function of V. We fix $(\lambda_1, \lambda_2) = (0.5, 1.0)$ (point J on the left panel of Fig. 6) and vary V between 1 and 1000. Recall that the worst-case

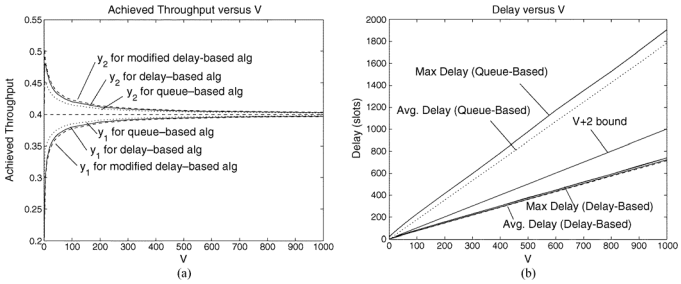


Fig. 7. (a) Achieved throughput and (b) delay performance versus V (for $(\lambda_1, \lambda_2) = (0.5, 1)$).

delay guarantee is $V + 2$ slots. Fig. 7(a) shows the resulting achieved throughputs versus V for the delay-based algorithm with known arrival rates, the modified delay-based algorithm with unknown arrival rates, and the queue-based algorithm. The achieved throughputs $(\overline{y}_1, \overline{y}_2)$ for all three algorithms are very close and converge to the optimal values (0.4, 0.4) as V is increased. Finally, Fig. 7(b) shows the average delay and maximum observed delay for all three algorithms. The average delays for the delay-based and modified delay-based algorithms are very close and cannot be distinguished on the plot. The maximum delays for these are also similar and are only slightly larger than the average delays, suggesting that oscillations are tightly centered on the average (see also related exponential attraction results for queue-based algorithms in [26]). The average and maximum delays for the queue-based algorithm are significantly larger. Thus, not only does our new delay-based approach provide worst-case delay guarantees, but it significantly reduces average delay as compared to the queue-based approach.

VII. Conclusion

We have established a delay-based policy for joint stability and utility optimization. The policy provides deterministic worst-case delay bounds, with total throughput utility that is inversely proportional to the delay guarantee. The Lyapunov optimization approach for this delay-based problem is significantly different from that of backlog-based policies. We believe these results add significantly to our understanding of network delay and delay-efficient control laws.

Appendix A
Proof of Lemma 4

Here we show that the given delay-based control algorithm maximizes the following expression, which is an expression that considers only the terms in the right-hand side of the drift bound in Lemma 2 that involve control variables:

$$V\mathbb{E}\left[\hat{g}\left(\boldsymbol{\gamma}(t)\right)|\boldsymbol{\Theta}(t)\right] - \sum_l Z_l(t)\mathbb{E}\left[D_l(t) + \gamma_l(t)|\boldsymbol{\Theta}(t)\right] + \sum_l H_l(t)\mathbb{E}\left[\mu_l(t) + D_l(t)|\boldsymbol{\Theta}(t)\right].$$

The $\gamma_l(t)$ terms appear separably in this drift expression, and hence they can be optimally chosen by observing $\boldsymbol{\Theta}(t)$ and maximizing $V\hat{g}(\boldsymbol{\gamma}) - \sum_l Z_l(t)\gamma_l(t)$ subject to $-1 \leq \gamma_l(t) \leq 1$ for all links l. This is precisely the first phase of the control algorithm. The remaining terms can be rearranged as (written without the conditional expectation for convenience)

$$\sum_l H_l(t)\mu_l(t) + \sum_l \left(H_l(t) - Z_l(t)\right) D_l(t). \tag{41}$$

Define $m_l(t)$ by

$$m_l(t) = \begin{cases} 1, & \text{if } H_l(t) > 0 \text{ and } H_l(t) \geq Z_l(t) \\ 0, & \text{otherwise.} \end{cases}$$

Recall that the $D_l(t)$ decision is made *after* the transmission decision and is constrained to be 0 unless a packet in queue l was not transmitted successfully. Regardless of the transmission decision, the $D_l(t)$ term in the above expression is maximized by selecting $D_l(t) = 1$ if $m_l(t) = 1$ and the packet in queue l was not transmitted successfully, and selecting $D_l(t) = 0$ otherwise. This is exactly the packet-dropping rule of phase 3 in the control algorithm. Using this dropping rule, we must have $D_l(t)m_l(t) = D_l(t)$ (if $D_l(t) = 0$ it is trivially true, and if $D_l(t) = 1$, then $m_l(t) = 1$ by the dropping rule).

It now suffices to choose an optimal transmission vector $\boldsymbol{x}(t)$, where $x_l(t) \in \{0, 1\}$. Recall that $\mu_l(t) = x_l(t)1_l(t)$, where $1_l(t)$ is an indicator function that is 1 if and only if a packet in link l was transmitted successfully. Using $D_l(t) = D_l(t)m_l(t)$, the expression (41) is thus

$$\sum_l H_l(t)x_l(t)1_l(t) + \sum_l \left(H_l(t) - Z_l(t)\right) D_l(t)m_l(t).$$

By adding and subtracting the same thing, this is written as

$$\begin{aligned}
&\sum_l H_l(t)x_l(t)1_l(t) + \sum_l \left(H_l(t) - Z_l(t)\right) D_l(t)x_l(t)m_l(t) \\
&\quad + \sum_l \left(H_l(t) - Z_l(t)\right) D_l(t)\left(1 - x_l(t)\right) m_l(t) \\
&\quad = \sum_l H_l(t)x_l(t)1_l(t) \\
&\qquad + \sum_l \left(H_l(t) - Z_l(t)\right)\left(1 - 1_l(t)\right) x_l(t)m_l(t) \\
&\qquad + \sum_l \left(H_l(t) - Z_l(t)\right)\left(1 - x_l(t)\right) m_l(t)
\end{aligned} \tag{42}$$

where we have used the following identities:

$$D_l(t)x_l(t)m_l(t) = \left(1 - 1_l(t)\right) x_l(t)m_l(t) \tag{43}$$

$$D_l(t)\left(1 - x_l(t)\right) m_l(t) = \left(1 - x_l(t)\right) m_l(t). \tag{44}$$

Identity (43) holds because if $x_l(t) = 1$ and $m_l(t) = 1$, then $D_l(t) = 1$ if and only if $1_l(t) = 0$ (that is, a packet for which $m_l(t) = 1$ is dropped if and only if it is transmitted unsuccessfully). Likewise, (44) holds because if $x_l(t) = 0$ and $m_l(t) = 1$, then $D_l(t) = 1$. Rearranging terms of (42) that involve control decisions yields

$$\sum_l x_l(t)1_l(t)\left[H_l(t) - m_l(t)\left(H_l(t) - Z_l(t)\right)\right]. \tag{45}$$

However, we have

$$H_l(t) - m_l(t)\left(H_l(t) - Z_l(t)\right) = \min\left[H_l(t), Z_l(t)\right].$$

This is because if $m_l(t) = 0$, then $H_l(t) \leq Z_l(t)$, and so $H_l(t) = \min[H_l(t), Z_l(t)]$. If $m_l(t) = 1$, then $H_l(t) \geq Z_l(t)$,

and so $Z_l(t) = \min[H_l(t), Z_l(t)]$. Plugging this identity into the expression (45) yields

$$\sum_l x_l(t)1_l(t)\min\left[H_l(t), Z_l(t)\right].$$

Taking conditional expectations of the above with respect to $\boldsymbol{\Theta}(t)$ yields

$$\mathbb{E}\left[\sum_l x_l(t)1_l(t)\min\left[H_l(t), Z_l(t)\right]|\boldsymbol{\Theta}(t)\right].$$

We seek a control rule that observes $\boldsymbol{S}(t)$ and $\boldsymbol{\Theta}(t)$ and chooses $\boldsymbol{x}(t) \in \mathcal{X}$, so that the above expression is maximized. Define $\chi(t) = [\boldsymbol{S}(t), \boldsymbol{\Theta}(t), \boldsymbol{x}(t)]$. By iterated expectations, the above expression is

$$\mathbb{E}\left[\mathbb{E}\left[\sum_l x_l(t)1_l(t)\min\left[H_l(t), Z_l(t)\right]|\chi(t)\right]|\boldsymbol{\Theta}(t)\right]$$
$$= \mathbb{E}\left[\sum_l x_l(t)\Psi_l\left(\boldsymbol{x}(t), \boldsymbol{S}(t)\right)\min\left[H_l(t), Z_l(t)\right]|\boldsymbol{\Theta}(t)\right]$$

where we have used the fact that $\mathbb{E}[1_l(t)|\chi(t)] = \Psi(\boldsymbol{x}(t), \boldsymbol{S}(t))$. The above expectation is thus minimized by observing the current $\boldsymbol{S}(t)$ and allocating $\boldsymbol{x}(t) \in \mathcal{X}$ according to phase 2 of the control algorithm.

APPENDIX B
PROOF OF LEMMA 1

Here, we prove the Lyapunov drift inequality of Lemma 1. Squaring the queue update equation for $Z_l(t)$ in (19) and noting that $\max[a, 0]^2 \leq a^2$ for any real number a gives

$$\begin{aligned}Z_l(t+1)^2 &\leq \left[Z_l(t) - \lambda_l + D_l(t) + \gamma_l(t)\right]^2\\ &= Z_l(t)^2 + \left(\gamma_l(t) + D_l(t) - \lambda_l\right)^2\\ &\quad - 2Z_l(t)\left[\lambda_l - D_l(t) - \gamma_l(t)\right].\end{aligned}$$

Summing the above over $l \in \{1, \ldots, L\}$ and dividing by 2 yields

$$\begin{aligned}&\frac{1}{2}\sum_{l=1}^{L}\left[Z_l(t+1)^2 - Z_l(t)^2\right]\\ &\quad\leq \frac{1}{2}\sum_{l=1}^{L}\left(\gamma_l(t) + D_l(t) - \lambda_l\right)^2\\ &\qquad - \sum_{l=1}^{L}Z_l(t)\left[\lambda_l - D_l(t) - \gamma_l(t)\right].\end{aligned} \tag{46}$$

Similarly, squaring (21) and noting that $\alpha_l(t) \in \{0, 1\}$ and $\beta_l(t) = 1 - \alpha_l(t)$ yields

$$\begin{aligned}H_l(t+1)^2 &\leq \alpha_l(t)\left[H_l(t)^2 + \left(1 - \left(\mu_l(t) + D_l(t)\right)T_l(t)\right)^2\right]\\ &\quad - \alpha_l(t)2H_l(t)\left[\left(\mu_l(t) + D_l(t)\right)T_l(t) - 1\right]\\ &\quad + \beta_l(t)A_l(t)^2\\ &= H_l(t)^2 + \alpha_l(t)\left(1 - \left(\mu_l(t) + D_l(t)\right)T_l(t)\right)^2\\ &\quad - 2H_l(t)\left[\left(\mu_l(t) + D_l(t)\right)T_l(t) - 1\right] + \beta_l(t)A_l(t)\end{aligned}$$

where the equality above uses the identities $\alpha_l(t)H_l(t) = H_l(t)$ and $A_l(t)^2 = A_l(t)$. Multiplying the above by $\lambda_l/2$ and summing over $l \in \{1, \ldots, L\}$ yields

$$\begin{aligned}&\frac{1}{2}\sum_{l=1}^{L}\lambda_l\left[H_l(t+1)^2 - H_l(t)^2\right]\\ &\quad\leq \frac{1}{2}\sum_{l=1}^{L}\lambda_l\left[\alpha_l(t)\left(1 - \left(\mu_l(t) + D_l(t)\right)T_l(t)\right)^2 + \beta_l(t)A_l(t)\right]\\ &\qquad - \sum_{l=1}^{L}\lambda_l H_l(t)\left[\left(\mu_l(t) + D_l(t)\right)T_l(t) - 1\right].\end{aligned} \tag{47}$$

Combining (46) and (47) and taking conditional expectations given the queue values $\boldsymbol{\Theta}(t)$ yields

$$\begin{aligned}\Delta(\boldsymbol{\Theta}(t)) &\leq \mathbb{E}[B(t)|\boldsymbol{\Theta}(t)] - \sum_l Z_l(t)\mathbb{E}[\lambda_l - D_l(t) - \gamma_l(t)|\boldsymbol{\Theta}(t)]\\ &\quad - \sum_l \lambda_l H_l(t)\mathbb{E}\left[\left(\mu_l(t) + D_l(t)\right)T_l(t) - 1|\boldsymbol{\Theta}(t)\right]\end{aligned}$$

where $B(t)$ is defined

$$\begin{aligned}B(t) &\triangleq \frac{1}{2}\sum_{l=1}^{L}\left[\left(\gamma_l(t) + D_l(t) - \lambda_l\right)^2 + \lambda_l\beta_l(t)A_l(t)\right]\\ &\quad + \frac{1}{2}\sum_{l=1}^{L}\lambda_l\alpha_l(t)\left[1 - \left(\mu_l(t) + D_l(t)\right)T_l(t)\right]^2.\end{aligned}$$

It remains only to show that $\mathbb{E}[B(t)|\boldsymbol{\Theta}(t)] \leq B$ for some finite constant B. Because $\gamma_l(t) \in [-1, 1]$, $D_l(t) + \mu_l(t) \in \{0, 1\}$, and $T_l(t) \geq 1$ if $\alpha_l(t) > 0$, we have

$$\left(\gamma_l(t) + D_l(t) - \lambda_l\right)^2 \leq \max\left[(2 - \lambda_l)^2, (\lambda_l + 1)^2\right]$$
$$\alpha_l(t)\left[1 - \left(\mu_l(t) + D_l(t)\right)T_l(t)\right]^2 \leq \alpha_l(t)T_l(t)^2.$$

Therefore

$$\begin{aligned}B(t) &\leq \frac{1}{2}\sum_{l=1}^{L}\max\left[(2 - \lambda_l)^2, (\lambda_l + 1)^2\right]\\ &\quad + \frac{1}{2}\sum_{l=1}^{L}\lambda_l\left[A_l(t) + \alpha_l(t)T_l(t)^2\right].\end{aligned}$$

Now, note that because $A_l(t)$ is i.i.d. over slots, it is independent of $\boldsymbol{\Theta}(t)$ and $\mathbb{E}[A_l(t)|\boldsymbol{\Theta}(t)] = \lambda_l$. Recall that $\alpha_l(t)$ is an indicator variable that is 1 if and only if $H_l(t) > 0$. Because we assume that the algorithm has the independence property, given $H_l(t) > 0$, $T_l(t)$ is independent of $\boldsymbol{\Theta}(t)$, and so

$$\begin{aligned}\mathbb{E}\left[\alpha_l(t)T_l(t)^2|\boldsymbol{\Theta}(t)\right] &= \mathbb{E}\left[T_l(t)^2|\boldsymbol{\Theta}(t), H_l(t) > 0\right]\\ &\quad \times \Pr\left[H_l(t) > 0|\boldsymbol{\Theta}(t)\right]\\ &\leq \mathbb{E}\left[T_l(t)^2|\boldsymbol{\Theta}(t), H_l(t) > 0\right]\\ &= 2/\lambda_l^2 - 1/\lambda_l\end{aligned}$$

where the final equality is the second moment of a geometric random variable with success probability λ_l. Therefore

$$\mathbb{E}\left[B(t)|\mathbf{\Theta}(t)\right] \leq \frac{1}{2}\sum_{l=1}^{L}\max\left[(2-\lambda_l)^2,(\lambda_l+1)^2\right] + \frac{1}{2}\sum_{l=1}^{L}\left[\lambda_l^2+\frac{2}{\lambda_l}-1\right].$$

Defining B as the right-hand side of the above inequality proves the result.

REFERENCES

[1] M. J. Neely, "Delay-based network utility maximization," in *Proc. IEEE INFOCOM*, Mar. 2010, pp. 1–9.

[2] L. Georgiadis, M. J. Neely, and L. Tassiulas, "Resource allocation and cross-layer control in wireless networks," *Found. Trends Netw.*, vol. 1, no. 1, pp. 1–149, 2006.

[3] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Trans. Autom. Control*, vol. 37, no. 12, pp. 1936–1948, Dec. 1992.

[4] L. Tassiulas and A. Ephremides, "Dynamic server allocation to parallel queues with randomly varying connectivity," *IEEE Trans. Inf. Theory*, vol. 39, no. 2, pp. 466–478, Mar. 1993.

[5] N. McKeown, A. Mekkittikul, V. Anantharam, and J. Walrand, "Achieving 100% throughput in an input-queued switch," *IEEE Trans. Commun.*, vol. 47, no. 8, pp. 1260–1267, Aug. 1999.

[6] P. R. Kumar and S. P. Meyn, "Stability of queueing networks and scheduling policies," *IEEE Trans. Autom. Control*, vol. 40, no. 2, pp. 251–260, Feb. 1995.

[7] E. Leonardi, M. Mellia, F. Neri, and M. A. Marsan, "Bounds on average delays and queue size averages and variances in input-queued cell-based switches," in *Proc. IEEE INFOCOM*, 2001, vol. 2, pp. 1095–1103.

[8] N. Kahale and P. E. Wright, "Dynamic global packet routing in wireless networks," in *Proc. IEEE INFOCOM*, 1997, vol. 3, pp. 1414–1421.

[9] M. Andrews, K. Kumaran, K. Ramanan, A. Stolyar, P. Whiting, and R. Vijaykumar, "Providing quality of service over a shared wireless link," *IEEE Commun. Mag.*, vol. 39, no. 2, pp. 150–154, Feb. 2001.

[10] M. J. Neely, E. Modiano, and C. E. Rohrs, "Dynamic power allocation and routing for time varying wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 1, pp. 89–103, Jan. 2005.

[11] M. Kobayashi, G. Caire, and D. Gesbert, "Impact of multiple transmit antennas in a queued SDMA/TDMA downlink," in *Proc. 6th IEEE SPAWC*, Jun. 2005, pp. 540–544.

[12] M. J. Neely and R. Urgaonkar, "Optimal backpressure routing in wireless networks with multi-receiver diversity," *Ad Hoc Netw.*, vol. 7, no. 5, pp. 862–881, July 2009.

[13] M. J. Neely, E. Modiano, and C. Li, "Fairness and optimal stochastic control for heterogeneous networks," in *Proc. IEEE INFOCOM*, Mar. 2005, vol. 3, pp. 1723–1734.

[14] M. J. Neely., "Dynamic power allocation and routing for satellite and wireless networks with time varying channels," Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA, 2003, LIDS.

[15] A. Stolyar, "Maximizing queueing network utility subject to stability: Greedy primal-dual algorithm," *Queueing Syst.*, vol. 50, no. 4, pp. 401–457, 2005.

[16] X. Lin and N. B. Shroff, "Joint rate control and scheduling in multihop wireless networks," in *Proc. 43rd IEEE Conf. Decision Control*, Paradise Island, Bahamas, Dec. 2004, vol. 2, pp. 1484–1489.

[17] A. Eryilmaz and R. Srikant, "Fair resource allocation in wireless networks using queue-length-based scheduling and congestion control," in *Proc. IEEE INFOCOM*, Mar. 2005, vol. 3, pp. 1794–1803.

[18] J. W. Lee, R. R. Mazumdar, and N. B. Shroff, "Opportunistic power scheduling for dynamic multiserver wireless systems," *IEEE Trans. Wireless Commun.*, vol. 5, no. 6, pp. 1506–1515, Jun. 2006.

[19] R. Agrawal and V. Subramanian, "Optimality of certain channel aware scheduling policies," in *Proc. 40th Annu. Allerton Conf. Commun., Control, Comput.*, Monticello, IL, Oct. 2002, pp. 1532–1541.

[20] H. Kushner and P. Whiting, "Asymptotic properties of proportional-fair sharing algorithms," in *Proc. 40th Annu. Allerton Conf. Commun., Control, Comput.*, 2002, pp. 1051–1059.

[21] A. Mekkittikul and N. McKeown, "A starvation free algorithm for achieving 100% throughput in an input-queued switch," in *Proc. ICCN*, 1996, pp. 226–231.

[22] M. Andrews, K. Kumaran, K. Ramanan, A. Stolyar, R. Vijaykumar, and P. Whiting, "Scheduling in a queueing system with asynchronously varying service rates," *Probab. Eng. Inf. Sci.*, vol. 18, no. 2, pp. 191–217, April 2004.

[23] S. Shakkottai and A. Stolyar, "Scheduling for multiple flows sharing a time-varying channel: The exponential rule," *Anal. Methods Appl. Probab., Amer. Math. Soc. Transl.*, vol. 207, pp. 185–202, 2002.

[24] R. Berry and R. Gallager, "Communication over fading channels with delay constraints," *IEEE Trans. Inf. Theory*, vol. 48, no. 5, pp. 1135–1149, May 2002.

[25] M. J. Neely, "Super-fast delay tradeoffs for utility optimal fair scheduling in wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 8, pp. 1489–1501, Aug. 2006.

[26] L. Huang and M. J. Neely, "Delay reduction via Lagrange multipliers in stochastic network optimization," *IEEE Trans. Autom. Control*, vol. 56, no. 4, pp. 842–857, Apr. 2011.

[27] D. P. Bertsekas and R. Gallager, *Data Networks*. Upper Saddle River, NJ: Prentice-Hall, 1992.

[28] M. J. Neely, *Stochastic Network Optimization With Application to Communication and Queueing Systems*. San Francisco, CA: Morgan & Claypool, 2010.

[29] B. Sadiq, S. Baek, and G. de Veciana, "Delay-optimal opportunistic scheduling and approximations: The log rule," in *Proc. IEEE INFOCOM*, Apr. 2009, pp. 1692–1700.

[30] M. J. Neely, "Stochastic optimization for Markov modulated networks with application to delay constrained wireless scheduling," in *Proc. IEEE CDC*, Shanghai, China, Dec. 2009, pp. 4826–4833.

[31] J. K. MacKie-Mason and H. R. Varian, "Pricing congestible network resources," *IEEE J. Sel. Areas Commun.*, vol. 13, no. 7, pp. 1141–1149, Sep. 1995.

[32] M. J. Neely and E. Modiano, "Convexity in queues with general inputs," *IEEE Trans. Inf. Theory*, vol. 51, no. 2, pp. 706–714, Feb. 2005.

[33] P. Giaccone, E. Leonardi, and D. Shah, "Throughput region of finite buffered networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 18, no. 2, pp. 251–263, Feb. 2007.

[34] M. J. Neely, "Energy optimal control for time varying wireless networks," *IEEE Trans. Inf. Theory*, vol. 52, no. 7, pp. 2915–2934, Jul. 2006.

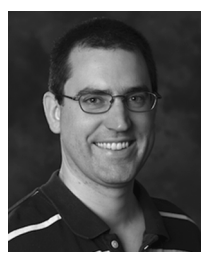
[35] L. B. Le, E. Modiano, and N. B. Shroff, "Optimal control of wireless networks with finite buffers," in *Proc. IEEE INFOCOM*, 2010, pp. 1–9.

[36] L. Ying, S. Shakkottai, A. Reddy, and S. Liu, "On combining shortest-path and back-pressure routing over multihop wireless networks," *IEEE/ACM Trans. Netw.*, vol. 19, no. 3, pp. 841–854, Jun. 2011.

[37] M. J. Neely, "Opportunistic scheduling with worst case delay guarantees in single and multi-hop networks," in *Proc. IEEE INFOCOM*, 2011, pp. 1728–1736.

[38] R. Daniels and R. W. Heath, Jr., "An online learning framework for link adaptation in wireless networks," in *Proc. Inf. Theory Appl. Workshop*, San Deigo, CA, Feb. 2009, pp. 138–140.

[39] F. Kelly, "Charging and rate control for elastic traffic," *Eur. Trans. Telecommun.*, vol. 8, no. 1, pp. 33–37, Jan.–Feb. 1997.



Michael J. Neely (S'00–M'02–SM'08) received the B.S. degrees in electrical engineering and mathematics from the University of Maryland, College Park, in 1997, and the M.S. and Ph.D. degrees in electrical engineering from the Massachusetts Institute of Technology (MIT), Cambridge, in 1999 and 2003, respectively.

In 2004, he joined the faculty of Electrical Engineering with the University of Southern California, Los Angeles, where he is currently an Associate Professor. His research is in the area of stochastic network optimization, with applications to wireless networks, mobile ad hoc networks, and switching systems.

Dr. Neely is a member of Tau Beta Pi and Phi Beta Kappa. He received the NSF Career Award in 2008 and the Viterbi School of Engineering Junior Research Award in 2009. He was awarded a three-year Department of Defense NDSEG Fellowship for graduate study at MIT.