Opportunistic Scheduling with Worst Case Delay Guarantees in Single and Multi-Hop Networks

Michael J. Neely

***Abstract*— We first consider a multi-user, single-hop wireless network with arbitrarily varying (and possibly non-ergodic) arrivals and channels. We design an opportunistic scheduling algorithm that guarantees all sessions have a bounded worst case delay. The algorithm has no knowledge of the future, but yields throughput-utility that is close to (or better than) that of a T-slot lookahead policy that makes "ideal" decisions based on perfect knowledge up to T slots into the future. We then extend the algorithm to treat worst case delay guarantees in multi-hop networks. Our analysis uses a sample-path version of Lyapunov optimization together with a novel virtual queue structure.**

***Index Terms*— Queueing analysis, optimization, flow control, wireless networks**

I. INTRODUCTION

This paper seeks to develop scheduling algorithms for wireless networks that: (i) provide efficient throughput-utility, (ii) are robust to general time-varying conditions, and (iii) guarantee bounded worst-case delay. While utility maximization is well studied for both static and stochastic networks, existing solutions make restrictive assumptions on the underlying stochastic processes and/or provide either no delay guarantees, or only weak guarantees on average delay. It is important to provide stronger delay guarantees to network users. This is particularly challenging for networks with unpredictable (possibly non-ergodic) traffic and channels. We overcome this challenge by integrating sample-path analysis and a novel virtual queue structure into the existing Lyapunov optimization framework for stochastic network optimization.

The theory of Lyapunov optimization yields algorithms for single-hop and multi-hop networks that provide throughput-utility within $O(\epsilon)$ of optimal (for any desired $\epsilon > 0$), with an $O(1/\epsilon)$ bound on average queue backlog [1]. However, these prior algorithms may yield unbounded worst-case delays, and can only provide bounds on *overall average delay* if traffic flows are "long-lived" and ergodic. This is done via Little's Theorem, which gives an average delay bound simply by dividing the average backlog bound by the total traffic rate λ_{tot}. This does not give bounds on the delays of individual sessions. Further, if some sessions are "short-lived" and consist only of a few packets, these packets may experience infinite delay due to the lack of "backpressure." Our recent work [2] considers *delay-based* rules for stochastic network optimization, and these do provide worst case delay guarantees. However, the work [2] assumes that packet arrivals of each user are independent and i.i.d. over slots, with known arrival rates λ_m for each user m. The analysis in [2] does not immediately generalize to non-i.i.d. situations or to multi-hop networks.

Related work on utility optimization for static networks can be found in [3][4][5][6], which solve for utility-optimal flows over the network. These do not consider queueing aspects or delays associated with randomly arriving traffic or time-varying channel conditions. Opportunistic scheduling rules for "infinitely backlogged" sources with time-varying and ergodic channels are considered in [7][8][9], but these do not consider queueing or delay. Lyapunov optimization approaches, as described in the preceding paragraph, are developed in [1][10][11][12], and are based on earlier works on Lyapunov drift for stability in [13][14][15][16][17]. Lyapunov optimization with finite buffers is treated in [12][18]. A Lyapunov optimization algorithm with a hop-count based queueing structure is used in [19] to provide a worst-case hop count to the destination, although this does not guarantee bounded delay. Network scheduling in a "fluid-flow" context, again mainly addressing ergodic stability without direct treatment of delay, is treated in [20][21][22][23]. Delay in small (single-queue) networks is treated with stochastic analysis in [24][25][26] and for systems with known future sample-paths in [27][28][29].

The next section describes a simple one-hop network model and presents the dynamic algorithm. Section III extends to multi-hop networks.

II. FORMULATION FOR ONE-HOP NETWORKS

Consider a network of M queues. The network operates in discrete time with unit timeslots $t \in \{0, 1, 2, \ldots\}$. Let $\boldsymbol{Q}(t) = (Q_1(t), \ldots, Q_M(t))$ represent the queue backlog in each queue on slot t. Depending on the context, the backlog can either take integer units of *packets* or real units of *bits*. Let $\boldsymbol{A}(t) = (A_1(t), \ldots, A_M(t))$ be the vector of new arrivals to each queue. Every slot, a network scheduler determines how much of the newly arriving data to admit, how much existing data in the queue to serve, and how much to drop. The queue dynamics for queues $m \in \{1, \ldots, M\}$ are given by:

$$Q_m(t+1) = \max[Q_m(t) - \mu_m(t) - D_m(t), 0] + R_m(t) \quad (1)$$

where $\mu_m(t)$ is the amount of type-m data offered service on slot t (also called the *service rate* for slot t, in data units per slot), $D_m(t)$ is the amount that can be dropped on slot t, and $R_m(t)$ is the amount admitted.

The admission decisions $R_m(t)$ are made every slot subject to the constraint $0 \leq R_m(t) \leq A_m(t)$, and any non-admitted data is treated as if it is dropped. The service rate vector $\boldsymbol{\mu}(t) = (\mu_1(t), \ldots, \mu_M(t))$ is determined by the current *channel condition* $S(t)$ and the current *resource allocation decision*

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

This material is supported in part under one or more of the following grants: DARPA IT-MANET W911NF-07-0028, NSF CAREER CCF-0747525, and continuing through participation in the Network Science Collaborative Technology Alliance sponsored by the U.S. Army Research Laboratory.

$\alpha(t)$. As in [1], the channel condition $S(t)$ can be a multi-dimensional quantity that specifies channel information for each of the M queues. The resource allocation decision $\alpha(t)$ can be a multi-dimensional decision that represents service and modulation choices for each queue m, and is made based on knowledge of the current $S(t)$. Specifically, the network controller observes the current $S(t)$ and chooses $\alpha(t)$ within some abstract set $\mathcal{A}_{S(t)}$ that specifies the decision options. The rates are then given by functions $\hat{\mu}_m(\alpha, S)$ as follows:

$$\mu_m(t) = \hat{\mu}_m(\alpha(t), S(t)) \;\; \forall m \in \{1, \ldots, M\}$$

We assume a maximum transmission rate μ_m^{max}, regardless of $\alpha(t), S(t)$, so that:

$$0 \leq \hat{\mu}_m(\alpha(t), S(t)) \leq \mu_m^{max}$$

Packet drop decisions $D_m(t)$ are made every slot subject to the constraints:

$$0 \leq D_m(t) \leq D_m^{max}$$

where for each $m \in \{1, \ldots, M\}$, D_m^{max} is a finite value that specifies the maximum amount of type-m data we are allowed to drop on one slot. We assume the channel state process $S(t)$ and the arrival vector $\boldsymbol{A}(t)$ have arbitrary sample paths. However, we assume arrivals have a finite maximum, so that $A_m(t) \leq A_m^{max}$ for all t and all $m \in \{1, \ldots, M\}$.

While the dynamic algorithm that we develop will base decisions on queue backlogs, it is useful to allow a general decision for $\alpha(t)$ and $D_m(t)$ to be decoupled from queue backlogs. Thus, we define $\tilde{\mu}_m(t)$ as the *actual* amount of type-m data served on slot t, defined by:

$$\tilde{\mu}_m(t) \triangleq \min[Q_m(t), \mu_m(t)]$$

Similarly, let $\tilde{D}_m(t)$ be the *actual* amount of type-m data dropped on slot t. The data is served (and dropped) in First-In-First-Out order. Any remaining data that is not served on slot t is subject to being dropped, with $\tilde{D}_m(t)$ defined by:

$$\tilde{D}_m(t) \triangleq \min[Q_m(t) - \tilde{\mu}_m(t), D_m(t)] \tag{2}$$

Clearly $\tilde{\mu}_m(t) \leq \mu_m(t)$ and $\tilde{D}_m(t) \leq D_m(t)$ for all m and t.

A. Discussion of Packet Drops

Here we distinguish between the packet drops $A_m(t) - R_m(t)$ due to transport layer admission decisions, and the packet drops $D_m(t)$ due to network layer queueing decisions. This distinction will be useful for distributed implementation of our multi-hop protocols in Section III. For both single and multi-hop networks, we strive to keep the rate at which the network layer drops data close to zero (only dropping when a delay constraint is violated). Thus, the bulk of packet drops take place at the transport layer, which acts to shape incoming traffic so that it is supportable on the network.

A complete network protocol requires an additional mechanism for retransmitting dropped packets. One simple mechanism is to simply include packet drops into the future packet arrivals. However, this creates an arrival process $A_m(t)$ that depends on the past control decisions. While we show that our analysis holds for arbitrary sample paths for arrivals, including those that are influenced by past control decisions, our comparison against a "genie aided" policy is the most meaningful if we assume the arrival process $A_m(t)$ is not influenced by past decisions. Without this assumption, all of our results still hold, but we emphasize that the "genie-aided" T-slot lookahead problem to be defined in (10)-(14) defines utility $util_k^*(T)$ in terms of the *actual* future arrivals $\boldsymbol{A}(\tau)$ experienced for $\tau \in \{kT, \ldots, kT + T - 1\}$, not in terms of the alternative values the arrivals might take if we were to choose different control actions.

B. Worst-Case Delay Via Persistent-Service Queues

The particular throughput-utility we consider is defined in the next subsection. Here, we develop a novel virtual queue, called an *ϵ-persistent service queue*, that can ensure bounded worst case delay for general types of utility functions. To this end, for each user $m \in \{1, \ldots, M\}$ define a virtual queue $Z_m(t)$ with initial backlog $Z_m(0) = 0$, and with queue update:

$$\begin{aligned} Z_m(t+1) \;\; = \;\; & \max[Z_m(t) + 1_{\{Q_m(t)>0\}}(\epsilon_m - \mu_m(t)) \\ & -D_m(t) - 1_{\{Q_m(t)=0\}}\mu_m^{max}, 0] \end{aligned} \tag{3}$$

where $\epsilon_m > 0$ are pre-specified constants, and $1_{\{Q_m(t)>0\}}$ is an indicator function that is 1 if $Q_m(t) > 0$, and 0 else. When $Q_m(t) > 0$, the virtual queue $Z_m(t)$ has the same departure process $\mu_m(t) + D_m(t)$ as $Q_m(t)$, but has a constant arrival of size ϵ_m. Any algorithm that maintains bounded $Z_m(t)$ and $Q_m(t)$ values also ensures persistent service with bounded worst-case delay, as shown in the next lemma.

Lemma 1: (Worst-Case Delay) Suppose an algorithm is used that ensures the following for all slots $t \in \{0, 1, 2, \ldots\}$:

$$Q_m(t) \leq Q_m^{max} \;\; , \;\; Z_m(t) \leq Z_m^{max}$$

where Q_m^{max} and Z_m^{max} are finite upper bounds on actual and virtual queue backlog. Then, assuming FIFO service, the worst-case delay of non-dropped data in queue m is bounded by the constant W_m^{max} defined below:

$$W_m^{max} \triangleq \lceil (Q_m^{max} + Z_m^{max})/\epsilon_m \rceil \tag{4}$$

where $\lceil x \rceil$ denotes the smallest integer that is greater than or equal to x. That is, all data that enters queue m is either served within W_m^{max} slots, or dropped.

Proof: Fix any slot $t \geq 0$, and let $A_m(t)$ represent the data that arrives on this slot. From (1), the earliest it can depart the queue is slot $t+1$. We show that all of this data departs (by being either served or dropped) on or before time $t + W_m^{max}$. Suppose this is *not* true. We shall reach a contradiction. It must be that $Q_m(\tau) > 0$ for all $\tau \in \{t+1, \ldots, t + W_m^{max}\}$ (else, we would clear the data by $t + W_m^{max}$). It follows from (3) that for all $\tau \in \{t+1, \ldots, t + W_m^{max}\}$ we have:

$$Z_m(\tau + 1) = \max[Z_m(\tau) + \epsilon_m - \mu_m(\tau) - D_m(\tau), 0]$$

and hence for all $\tau \in \{t+1, \ldots, t + W_m^{max}\}$:

$$Z_m(\tau + 1) \geq Z_m(\tau) + \epsilon_m - \mu_m(\tau) - D_m(\tau)$$

Summing the above over $\tau \in \{t+1, \ldots, t + W_m^{max}\}$ yields:

$$\begin{aligned} & Z_m(t + W_m^{max} + 1) - Z_m(t+1) \\ & \quad \geq \epsilon_m W_m^{max} - \textstyle\sum_{\tau=t+1}^{t+W_m^{max}} [\mu_m(\tau) + D_m(\tau)] \end{aligned}$$

Rearranging terms in the above inequality and using the fact that $Z_m(t+W_m^{max}+1) \leq Z_m^{max}$ and $Z_m(t+1) \geq 0$ yields:

$$\epsilon_m W_m^{max} - Z_m^{max} \leq \sum_{\tau=t+1}^{t+W_m^{max}} [\mu_m(\tau) + D_m(\tau)] \quad (5)$$

Because service is FIFO, the data $A_m(t)$ that arrives on slot t is placed at the end of the queue on slot $t+1$ (see queue dynamics (1)), and this data is fully cleared only when all of the backlog $Q_m(t+1)$ has departed. That is, the last of the $A_m(t)$ data departs on the slot $t+T$, where $T > 0$ is the smallest integer for which $\sum_{\tau=t+1}^{t+T}[\mu_m(\tau) + D_m(\tau)] \geq Q_m(t+1)$. Because we have assumed that not all of the $A_m(t)$ data departs by time $t+W_m^{max}$, we must have:

$$\sum_{\tau=t+1}^{t+W_m^{max}} [\mu_m(\tau) + D_m(\tau)] < Q_m(t+1) \leq Q_m^{max} \quad (6)$$

Combining (6) and (5) yields:

$$\epsilon_m W_m^{max} - Z_m^{max} < Q_m^{max}$$

Therefore:

$$W_m^{max} < (Q_m^{max} + Z_m^{max})/\epsilon_m$$

This contradicts the definition of W_m^{max} given in (4). □

Our approach now is to design a utility-efficient algorithm that maintains finite Q_m^{max} and Z_m^{max} for all users m.

C. Utility Functions and Intuition from Ergodic Systems

As a measure of throughput-utility, for each user $m \in \{1, \ldots, M\}$ define $g_m(r)$ as a continuous, concave, and non-decreasing function over the interval $0 \leq r \leq A_m^{max}$. Assume that each utility function $g_m(r)$ has a finite maximum slope of ν_m (being the right-derivative of $g_m(r)$ at $r = 0$). An example utility function with maximum slope ν_m is given by:

$$g_m(r) \triangleq \nu_m \theta \log(1 + r/\theta)$$

where $\theta > 0$ is any real number and $\log(\cdot)$ denotes the natural logarithm.

For intuition, suppose (temporarily) that we have an ergodic setting, and we use control algorithms for which all time averages exist. Define $\overline{r}_m$ as the time average value of $R_m(t)$ and as $t \to \infty$. Similarly define $\overline{d}_m$ as the time average of drops $D_m(t)$ at queue m, and $\overline{\mu}_m$ as the time average transmission rate $\mu_m(t)$ at node m. Note that in a system with stable queues, the value $\overline{r}_m - \overline{d}_m$ represents the *throughput* of the system. Then one may seek to design a scheduling and packet dropping algorithm that solves the following:

$$\begin{aligned} &\text{Maximize:} && \textstyle\sum_{m=1}^{M} g_m(\overline{r}_m - \overline{d}_m) \\ &\text{Subject to:} && \overline{r}_m \leq \overline{\mu}_m + \overline{d}_m \;\; \forall m \in \{1, \ldots, M\} \end{aligned}$$

It is not difficult to show that an optimal solution of the above problem can be found with $\overline{d}_m = 0$ for all m. That is because any data that was dropped at the network layer could just as easily have been dropped during the admission decisions.[1]

[1]This observation is particularly important in the multi-hop setting, where it is preferable to drop data at the transport layer rather than suffering the inefficiencies of transmitting data in the network layer only to drop it before it reaches the destination.

Further, it is easy to show that for any value $\beta \geq 1$, the above problem is equivalent to the following:

$$\text{Maximize:} \quad \sum_{m=1}^{M}[g_m(\overline{r}_m) - \beta\nu_m\overline{d}_m] \quad (7)$$

$$\text{Subject to:} \quad \overline{r}_m \leq \overline{\mu}_m + \overline{d}_m \;\; \forall m \in \{1, \ldots, M\} \quad (8)$$

This equivalence is because the value $\beta\nu_m$, being the slope of the cost function for packet drops $\overline{d}_m$, is greater than or equal to the maximum slope ν_m of the utility function $g_m(r)$. Therefore, admitting an extra unit of data (which improves the $g_m(\overline{r}_m)$ utility) and then dropping it later (which counts in the $\beta\nu_m\overline{d}_m$ cost) is no better than not admitting the extra unit in the first place. The above transformed problem has admission variables $\overline{r}_m$ and drop variables $\overline{d}_m$ that appear separably in the utility function, which will be useful in extended formulations for multi-hop networks.

We can solve problems of the type (7)-(8) using stochastic network optimization theory [1][30], and the solutions would never drop in-network data (so that $\overline{d}_m = 0$). In particular, such algorithms would stabilize all queues (with finite average backlog) while getting a utility that can be pushed arbitrarily close to the optimal utility of (7)-(8). However, such an approach cannot guarantee *finite worst-case delay*. We shall design an algorithm that uses in-network packet drops to ensure finite worst-case delay. Further, in ergodic settings, our new algorithm can guarantee an overall utility that is close to the optimal utility of the problem (7)-(8), *augmented with the following additional constraints*:

$$\overline{\mu}_m + \overline{d}_m \geq \epsilon_m \; \forall m \in \{1, \ldots, M\} \quad (9)$$

for some values $\epsilon_m > 0$.

In general, the optimal utility for the problem (7)-(9) is *less than or equal* to the optimal utility for the problem (7)-(8). However, if an optimal solution to (7)-(8) can be found that uses rates $\overline{r}_m^{opt}$ that satisfy $\overline{r}_m^{opt} \geq \epsilon_m$ for all m, then the additional constraints (9) are also satisfied by this solution so that these constraints *do not reduce utility*. This is typical when the ϵ_m values are small enough so that the optimal flow rates $\overline{r}_m^{opt}$ are all at least ϵ_m. Of course, in general situations the additional constraints (9) may reduce utility, but by no more than an amount $O(\epsilon)$, where $\epsilon \triangleq \max_{m \in \{1,\ldots,M\}} \epsilon_m$.

D. The T-Slot Lookahead Utility for General Sample Paths

While the algorithm of this paper achieves the performance described in the previous subsection in the special case of ergodic settings (such as when the arrival vector $\boldsymbol{A}(t)$ and the network channels $S(t)$ are i.i.d. over slots), we are interested in the more complex case of general sample paths (possibly those from non-ergodic processes). Thus, we cannot assume the network reaches an "equilibrium" with well defined time averages. We thus evaluate our algorithm against a *T-slot lookahead metric*.

For a given $S(t)$ on slot t, define the set $\mathcal{B}(S(t))$ as the convex hull of all rate vectors $\hat{\boldsymbol{\mu}}(\alpha, S(t)) = (\hat{\mu}_1(\alpha, S(t)), \ldots, \hat{\mu}_M(\alpha, S(t)))$ that can be achieved by resource allocation decisions $\alpha \in \mathcal{A}_{S(t)}$:

$$\mathcal{B}(S(t)) \triangleq Conv\{\boldsymbol{\mu} = \hat{\boldsymbol{\mu}}(\alpha, S(t)) | \alpha \in \mathcal{A}_{S(t)}\}$$

Now fix integers $T > 0$ and $K > 0$, and consider the first KT slots $\tau \in \{0, 1, \ldots, KT-1\}$. Decompose this horizon into K successive frames of size T. For each frame $k \in \{0, 1, \ldots, K-1\}$, consider the following optimization problem, which chooses constants d_m and values $\boldsymbol{\mu}(\tau)$, $R_m(\tau)$ over the frame $\tau \in \{kT, \ldots, kT+T-1\}$ based on full knowledge of the future $\boldsymbol{A}(\tau)$ and $S(\tau)$ values over this frame:

$$\text{Maximize:} \quad \sum_{m=1}^{M}\left[g_m\left(\frac{1}{T}\sum_{\tau=kT}^{kT+T-1}R_m(\tau)\right) - \beta\nu_m d_m\right] \quad (10)$$

Subject to:

$$\sum_{\tau=kT}^{kT+T-1}[R_m(\tau) - \mu_m(\tau) - d_m] \leq 0 \;\; \forall m \quad (11)$$

$$\frac{1}{T}\sum_{\tau=kT}^{kT+T-1}[\mu_m(\tau) + d_m] \geq \epsilon_m \;\; \forall m \quad (12)$$

$$0 \leq d_m \leq D_m^{max} \;,\; 0 \leq R_m(\tau) \leq A_m(\tau) \;\; \forall m, \tau \quad (13)$$

$$\boldsymbol{\mu}(\tau) = (\mu_1(\tau), \ldots, \mu_M(\tau)) \in \mathcal{B}(S(\tau)) \;\; \forall \tau \quad (14)$$

The above problem is an analogue of the problem (7)-(9) (note that (10), (11), (12) are analogues of (7), (8), (9), respectively). We assume throughout that $0 \leq \epsilon_m \leq D_m^{max}$ for all m, in which case it is easy to show that the constraints (11)-(14) are always achievable by the trivial strategy with $d_m = \epsilon_m$, $R_m(\tau) = 0$ for all τ. Define $util_k^*(T)$ as the supremum utility value of (10) in the above problem for frame k, and d_m^*, $\boldsymbol{\mu}^*(\tau)$, $R_m^*(\tau)$ as the corresponding decisions that achieve this optimal value.[2] The $util_k^*(T)$ value represents the optimum sum utility over frame k (using a frame size T) that can be achieved under the assumption that the sum arrivals over the frame are less than or equal to the sum departures, and that the empirical departure rate over the frame for each queue m is at least ϵ_m. The $\boldsymbol{\mu}^*(\tau)$ and d_m^* values cannot be used in practice because they would require full knowledge of the future for all slots in the frame. Further, $\boldsymbol{\mu}^*(\tau)$ is allowed to take values in the extended set $\mathcal{B}(S(\tau))$, being the convex hull of the set of actual transmission rates available on slot τ.

We shall design an algorithm that does not know the future, and that does not use the T parameter, but that achieves bounded worst case delay with an empirical average utility over the first KT slots (for all positive integers K and T) that is close to (or greater than) the value:

$$\frac{1}{K}\sum_{k=0}^{K-1} util_k^*(T)$$

The above value represents the average of the optimal utilities over each frame. While this does *not necessarily* represent the optimal utility that can be achieved over the full KT slots, it is still a meaningful value for comparison because it involves ideal decisions that have knowledge of the future up to T slots. Further, it can be shown that in ergodic settings:

$$\lim_{T\to\infty}\left[\lim_{K\to\infty}\frac{1}{K}\sum_{k=0}^{K-1} util_k^*(T)\right] = util^{opt}$$

where $util^{opt}$ is the optimal infinite horizon utility that can be achieved over all algorithms that stabilize the queues and provide a long-term $\mu_m(t) + D_m(t)$ rate greater than ϵ_m. The intuition for this is that in ergodic settings with T very large, the fraction of time different events occur over the frame approaches the ergodic distribution, so that $util_k^*(T)$ is close to $util^{opt}$ on every frame.

[2] For simplicity, we assume the supremum utility $util_k^*(T)$ for the problem (10)-(14) can be achieved. Else, we can recover the same results by taking a limit over decisions that approach $util_k^*(T)$.

E. Lyapunov Optimization and the Dynamic Algorithm

For each $m \in \{1, \ldots, M\}$, define an *auxiliary variable* $\gamma_m(t)$ and a *flow state queue* $Y_m(t)$, being a virtual queue with dynamics as follows:

$$Y_m(t+1) = \max[Y_m(t) - R_m(t) + \gamma_m(t), 0] \quad (15)$$

The auxiliary variable $\gamma_m(t)$ is chosen every slot in the interval $[0, A_m^{max}]$ in order to stabilize all queues while striving to maximize $\sum_m [g_m(\overline{\gamma}_m) - \beta\nu_m \overline{d}_m]$. The intuition is that stabilizing the $Y_m(t)$ queue ensures the time average of $R_m(t)$ is at least as large as the time average of $\gamma_m(t)$, and so the throughput-utility will also be large.

To this end, define $\boldsymbol{\Theta}(t) \triangleq [\boldsymbol{Q}(t), \boldsymbol{Z}(t), \boldsymbol{Y}(t)]$ as a collective vector of all $Q_m(t)$, $Z_m(t)$, and $Y_m(t)$ queues. Define the Lyapunov function $L(\boldsymbol{\Theta}(t))$ as follows:

$$L(\boldsymbol{\Theta}(t)) \triangleq \frac{1}{2}\sum_{m=1}^{M}[Q_m(t)^2 + Z_m(t)^2 + Y_m(t)^2]$$

Define $\Delta_T(\boldsymbol{\Theta}(t)) \triangleq L(\boldsymbol{\Theta}(t+T)) - L(\boldsymbol{\Theta}(t))$ as the T-step sample path Lyapunov drift. We have the following result for $T = 1$, being a bound on the 1-slot "drift-plus-penalty" value used in Lyapunov optimization [1][30]. The "penalty" $-\sum_{m=1}^{M}[g_m(\gamma_m(t)) - \beta\nu_m D_m(t)]$ is multiplied by a parameter $V > 0$ that will affect a performance-delay tradeoff.

Lemma 2: (1-Slot Drift) For all t, all $\boldsymbol{\Theta}(t)$, and any scheduling decisions made on slot t, we have:

$$\begin{aligned}\Delta_1(\boldsymbol{\Theta}(t)) - V\textstyle\sum_{m=1}^{M}[g_m(\gamma_m(t)) - \beta\nu_m D_m(t)] \leq \\ B - \textstyle\sum_{m=1}^{M} Z_m(t) 1_{\{Q_m(t)=0\}}\mu_m^{max} \\ -V\textstyle\sum_{m=1}^{M}[g_m(\gamma_m(t)) - \beta\nu_m D_m(t)] \\ +\textstyle\sum_{m=1}^{M} Q_m(t)[R_m(t) - \hat{\mu}_m(\alpha(t), S(t)) - D_m(t)] \\ +\textstyle\sum_{m=1}^{M} Z_m(t) 1_{\{Q_m(t)>0\}}[\epsilon_m - \hat{\mu}_m(\alpha(t), S(t))] \\ -\textstyle\sum_{m=1}^{M} Z_m(t) D_m(t) + \sum_{m=1}^{M} Y_m(t)[\gamma_m(t) - R_m(t)] \quad (16)\end{aligned}$$

where the constant B is defined:

$$\begin{aligned}B \triangleq \frac{1}{2}\textstyle\sum_{m=1}^{M}[(\mu_m^{max} + D_m^{max})^2 + 2(A_m^{max})^2] \\ + \frac{1}{2}\textstyle\sum_{m=1}^{M}\max[\epsilon_m^2, (\mu_m^{max} + D_m^{max})^2] \quad (17)\end{aligned}$$

Proof: See Appendix A. □

The following algorithm makes $\alpha(t)$, $R_m(t)$, $D_m(t)$, $\gamma_m(t)$ decisions every slot t to minimize the right-hand-side of the above drift-plus-penalty expression. Every slot t, observe $S(t)$ and the current queue states $\boldsymbol{\Theta}(t) = [\boldsymbol{Q}(t), \boldsymbol{Z}(t), \boldsymbol{Y}(t)]$. Then:

- (Transmission) Choose $\alpha(t) \in \mathcal{A}_{S(t)}$ to maximize the following expression:

$$\sum_{m=1}^{M}[Q_m(t) + Z_m(t) 1_{\{Q_m(t)>0\}}]\hat{\mu}_m(\alpha(t), S(t)) \quad (18)$$

- (Admission) Choose $R_m(t)$ by:

$$R_m(t) = \begin{cases} A_m(t) & \text{if } Q_m(t) \leq Y_m(t) \\ 0 & \text{otherwise} \end{cases} \quad (19)$$

- (Dropping) For each $m \in \{1, \ldots, M\}$, choose:

$$D_m(t) = \begin{cases} D_m^{max} & \text{if } Q_m(t) + Z_m(t) > V\beta\nu_m \\ 0 & \text{else} \end{cases} \quad (20)$$

- (Auxiliary Variables) Choose $\gamma_m(t)$ to solve:

$$\text{Maximize:} \quad Vg_m(\gamma_m(t)) - Y_m(t)\gamma_m(t) \tag{21}$$
$$\text{Subject to:} \quad 0 \leq \gamma_m(t) \leq A_m^{max} \tag{22}$$

- (Queue Updates) Update queues $Q_m(t)$, $Z_m(t)$, $Y_m(t)$ according to (1), (3), (15) (using $D_m(t)$, $\mu_m(t)$, $\gamma_m(t)$, $R_m(t)$ obtained above for the updates).

While $D_m(t)$, $\mu_m(t)$ are used for the queue updates (*not* the $\tilde{D}_m(t)$ and $\tilde{\mu}_m(t)$ values), there may be cases when $D_m(t) + \mu_m(t) > Q_m(t)$, so the same queue updates can be implemented by either serving as much data as possible on slot t and then dropping the rest, or dropping as much as possible and then serving the rest.[3] Both decision rules result in the same queue sample paths, and so our analytical bounds apply to both. However, it is clearly better to serve and then drop the remaining, so that the *actual* amount dropped from the physical queue is $\tilde{D}_m(t)$ given in (2).

F. Bounded Queues

We now show the above algorithm ensures queues are bounded by constants Y_m^{max}, Q_m^{max}, Z_m^{max} given by:

$$Y_m^{max} \triangleq V\nu_m + A_m^{max} \tag{23}$$
$$Q_m^{max} \triangleq V\nu_m + 2A_m^{max} \tag{24}$$
$$Z_m^{max} \triangleq V\beta\nu_m + \epsilon_m \tag{25}$$

Lemma 3: (Bounded Queues) Assume the decisions $R_m(t)$, $D_m(t)$, $\gamma_m(t)$, and queue updates are done according to the above dynamic algorithm, but the $\alpha(t)$ decisions use *any* algorithm (possibly different than the ones that maximize (18)). Assume $0 \leq \epsilon_m \leq D_m^{max}$ for all $m \in \{1, \ldots, M\}$. Then for all slots $t \geq 0$ and all $m \in \{1, \ldots, M\}$ we have:

$$Y_m(t) \leq Y_m^{max} \ , \ Q_m(t) \leq Q_m^{max} \ , \ Z_m(t) \leq Z_m^{max}$$

provided that these inequalities are satisfied at time $t = 0$.

Proof: (Lemma 3) We first prove that $Y_m(t) \leq Y_m^{max}$ for all t. Suppose this is true for some time t. We show it also holds for $t+1$. If $Y_m(t) \leq V\nu_m$, then $Y_m(t+1) \leq V\nu_m + A_m^{max} \triangleq Y_m^{max}$, because it can increase by at most A_m^{max} on any slot (see dynamics (15)). Now consider the opposite case where $Y_m(t) > V\nu_m$. Because ν_m is the maximum derivative of the $g_m(r)$ function, we have:

$$g_m(\gamma) \leq g_m(0) + \nu_m\gamma \quad \text{for } 0 \leq \gamma \leq A_m^{max}$$

with equality if $\gamma \leq 0$. Therefore, for all $\gamma_m(t)$ in the interval $0 \leq \gamma_m(t) \leq A_m^{max}$ we have:

$$\begin{aligned} & Vg_m(\gamma_m(t)) - Y_m(t)\gamma_m(t) \\ & \quad \leq \quad Vg_m(0) + V\nu_m\gamma_m(t) - Y_m(t)\gamma_m(t) \\ & \quad = \quad Vg_m(0) + \gamma_m(t)[V\nu_m - Y_m(t)] \leq Vg_m(0) \end{aligned}$$

with equality holding only if $\gamma_m(t) = 0$ (because the term $[V\nu_m - Y_m(t)]$ is negative). It follows in this case that the auxiliary variable decision rule (21)-(22) chooses $\gamma_m(t) = 0$ whenever $Y_m(t) > V\nu_m$, and so the $Y_m(t)$ queue cannot increase on the next slot (see dynamics (15)). Thus, $Y_m(t+1) \leq Y_m(t) \leq Y_m^{max}$, proving that the Y_m^{max} bound again holds. This proves $Y_m(t) \leq Y_m^{max}$ for all t.

We now prove that $Q_m(t) \leq Q_m^{max}$ for all t. Assume this is true for a particular slot t. We prove it also holds for slot $t+1$. If $Q_m(t) \leq Y_m^{max}$, then $Q_m(t+1) \leq Y_m^{max} + A_m^{max} \triangleq Q_m^{max}$ (because it can increase by at most A_m^{max} on one slot). Consider now the opposite case when $Q_m(t) > Y_m^{max}$. We thus have $Q_m(t) > Y_m^{max}$, and the admission decision (19) will choose $R_m(t) = 0$. Thus, the queue cannot increase and we have $Q_m(t+1) \leq Q_m(t) \leq Q_m^{max}$. This proves the Q_m^{max} bound. The proof of the Z_m^{max} bound is similar: If $Z_m(t) \leq V\beta\nu_m$, then $Z_m(t+1) \leq V\beta\nu_m + \epsilon_m \triangleq Z_m^{max}$. Else, the dropping rule (20) will choose $D_m(t) = D_m^{max}$, so that the queue cannot increase on the next slot. □

G. Algorithm Performance

The following theorem assumes that $0 \leq \epsilon_m \leq D_m^{max}$ for all m, and that $Q_m(0) = Z_m(0) = Y_m(0) = 0$ for all m.

Theorem 1: Under the above assumptions, if the above dynamic algorithm is used every slot t with a fixed parameter $V > 0$, then:

(a) Worst case queue backlog is bounded by Q_m^{max}, and worst case delay is bounded by W_m^{max}, where:

$$W_m^{max} \triangleq \lceil (Q_m^{max} + Z_m^{max})/\epsilon_m \rceil$$

where Q_m^{max} and Z_m^{max} are defined in (24) and (25). Note that $Q_m^{max} \leq O(V)$ and $W_m^{max} \leq O(V)$, so that worst case queue backlog and delay grow linearly in the V parameter.

(b) For any positive integers $K > 0$, $T > 0$, the throughput-utility over the first KT slots satisfies:

$$\begin{aligned} \textstyle\sum_{m=1}^M [g_m(\overline{r}_m(KT)) - \beta\nu_m\overline{d}_m(KT)] \geq \\ \tfrac{1}{K}\textstyle\sum_{k=0}^{K-1} util_k^*(T) - \tfrac{BT}{V} - \tfrac{1}{KT}\textstyle\sum_{m=1}^M \nu_m Y_m^{max} \end{aligned}$$

where the constant B is defined in (17), $util_k^*(T)$ is the optimal utility associated with the T-slot lookahead problem (10)-(14) for frame size T and frame k, Y_m^{max} is defined in (23), and $\overline{r}_m(KT)$, $\overline{d}_m(KT)$ are defined:

$$\overline{r}_m(KT) \triangleq \frac{1}{KT}\sum_{\tau=0}^{KT-1} R_m(\tau) \ , \ \overline{d}_m(KT) \triangleq \frac{1}{KT}\sum_{\tau=0}^{KT-1} D_m(\tau)$$

(c) If the combined process $[\boldsymbol{A}(t), S(t)]$ is i.i.d. over slots then with probability 1:

$$\liminf_{t\to\infty} \textstyle\sum_{m=1}^M [g_m(\overline{r}_m(t)) - \beta\nu_m\overline{d}_m(t)] \geq util^* - B/V$$

where B is defined in (17), and $util^*$ is the infinite horizon optimal utility for problem (7)-(9).

Note that the utility bound in part (b) can be simplified if we take a limit as $K \to \infty$:

$$\begin{aligned} \liminf_{K\to\infty} \textstyle\sum_{m=1}^M [g_m(\overline{r}_m(KT)) - \beta\nu_m\overline{d}_m(KT)] \geq \\ \liminf_{K\to\infty} \tfrac{1}{K}\textstyle\sum_{k=0}^{K-1} util_k^*(T) - \tfrac{BT}{V} \end{aligned}$$

This demonstrates that the limiting achieved utility is no less than $O(1/V)$ distance below the target utility. Thus, the V parameter ensures this distance can be made arbitrarily small, at the expense of an $O(V)$ tradeoff in worst case queue backlog and delay. The utility bounds above can be viewed as

[3] These cases become increasingly rare as V is increased.

a class of bounds that hold for all $T > 0$. If the T parameter is increased, $util_k^*(T)$ typically increases, at the expense of requiring a larger V parameter to make BT/V small.

H. Proof of Theorem 1

The queue bounds in part (a) follow from Lemma 3, and the worst case delay W_m^{max} follows immediately from these bounds together with Lemma 1. To prove part (b), we use the following lemma.

Lemma 4: (T-slot Drift) Under the above dynamic algorithm, for all t, all $\boldsymbol{\Theta}(t)$, and for any integer $T > 0$ we have:

$$\begin{aligned}\Delta_T(\boldsymbol{\Theta}(t)) - V\sum_{\tau=t}^{t+T-1}\sum_{m=1}^{M}[g_m(\gamma_m(\tau)) - \beta\nu_m D_m(\tau)] \leq \\ BT^2 - VT\sum_{m=1}^{M}[g_m(\gamma_m^*) - \beta\nu_m d_m^*] \\ + \sum_{m=1}^{M} Q_m(t)\sum_{\tau=t}^{t+T-1}[R_m^*(\tau) - \mu_m^*(\tau) - d_m^*] \\ + \sum_{m=1}^{M} Z_m(t)\sum_{\tau=t}^{t+T-1}[\epsilon_m - \mu_m^*(\tau) - d_m^*] \\ + \sum_{m=1}^{M} Y_m(t)\sum_{\tau=t}^{t+T-1}[\gamma_m^* - R_m^*(\tau)]\end{aligned}$$

where B is defined in (17) and γ_m^*, d_m^*, $R_m^*(\tau)$, $\boldsymbol{\mu}^*(\tau)$ are any values that satisfy $0 \leq \gamma_m^* \leq A_m^{max}$, $0 \leq d_m^* \leq D_m^{max}$, $0 \leq R_m^*(\tau) \leq A_m(\tau)$, and $\boldsymbol{\mu}^*(\tau) \in \mathcal{B}(S(\tau))$.

Proof: Omitted for brevity. □

Now consider slot $t = kT$, and define d_m^*, $R_m^*(\tau)$, $\boldsymbol{\mu}^*(\tau)$ (for $\tau \in \{kT, \ldots, (k+1)T - 1\}$) as the decisions that solve the T-slot lookahead problem (10)-(14) over frame k. Define $\gamma_m^* \triangleq \frac{1}{T}\sum_{\tau=kT}^{kT+T-1} R_m^*(\tau)$. Plugging this into the right-hand-side of the drift bound in Lemma 4 yields:

$$\begin{aligned}\Delta_T(\boldsymbol{\Theta}(kT)) - V\sum_{\tau=kT}^{kT+T-1}\sum_{m=1}^{M}[g_m(\gamma_m(\tau)) - \beta\nu_m D_m(\tau)] \leq \\ BT^2 - VTutil_k^*\end{aligned}$$

Summing the above over $k \in \{0, \ldots, K-1\}$ yields:

$$\begin{aligned}L(\boldsymbol{\Theta}(KT)) - L(\boldsymbol{\Theta}(0)) \\ -V\textstyle\sum_{\tau=0}^{KT-1}\sum_{m=1}^{M}[g_m(\gamma_m(\tau)) - \beta\nu_m D_m(\tau)] \leq \\ BKT^2 - VT\textstyle\sum_{k=0}^{K-1} util_k^*\end{aligned}$$

Rearranging terms and using the fact that $L(\boldsymbol{\Theta}(KT)) \geq 0$ and $L(\boldsymbol{\Theta}(0)) = 0$ yields:

$$\begin{aligned}\tfrac{1}{KT}\textstyle\sum_{\tau=0}^{KT-1}\sum_{m=1}^{M} g_m(\gamma_m(\tau)) - \sum_{m=1}^{M}\beta\nu_m\overline{d}_m(KT) \\ \geq \tfrac{1}{K}\textstyle\sum_{k=0}^{K-1} util_k^* - BT/V\end{aligned}$$

Using Jensen's inequality in the left-hand-side for the concave function $g_m(\cdot)$ yields:

$$\begin{aligned}\textstyle\sum_{m=1}^{M}[g_m(\overline{\gamma}_m(KT)) - \beta\nu_m\overline{d}_m(KT)] \geq \\ \tfrac{1}{K}\textstyle\sum_{k=0}^{K-1} util_k^* - BT/V\end{aligned} \quad (26)$$

where $\overline{\gamma}_m(KT)$ is the time average of $\gamma_m(\tau)$ over the first KT slots. However, from (15) we have for all τ:

$$Y_m(\tau+1) \geq Y_m(\tau) - R_m(\tau) + \gamma_m(\tau)$$

Summing over $\tau \in \{0, \ldots, KT-1\}$ and dividing by KT yields:

$$\begin{aligned}\frac{Y_m(KT) - Y_m(0)}{KT} &\geq -\frac{1}{KT}\sum_{\tau=0}^{KT} R_m(\tau) + \frac{1}{KT}\sum_{\tau=0}^{KT}\gamma_m(\tau) \\ &= -\overline{r}_m(KT) + \overline{\gamma}_m(KT)\end{aligned}$$

Using the fact that $Y_m(0) = 0$ and $Y_m(KT) \leq Y_m^{max}$ yields:

$$\overline{r}_m(KT) \geq \overline{\gamma}_m(KT) - Y_m^{max}/(KT)$$

Using the fact that $g_m(\gamma)$ is non-decreasing with largest derivative ν_m, from the above inequality we obtain:

$$g_m(\overline{r}_m(KT)) \geq g_m(\overline{\gamma}_m(KT)) - \nu_m Y_m^{max}/(KT)$$

Using the above in (26) proves part (b) of Theorem 1. Part (c) omitted for brevity.

III. Multi-Hop Networks

Consider now a multi-hop network with N nodes. As before, there are M traffic sessions with random arrivals $(A_1(t), \ldots, A_M(t))$. Each session m has a particular source node $source(m)$ and destination node $dest(m)$. Let $Q_n^{(m)}(t)$ represent the current amount of type-m data in node n. Let $\mu_{ab}^{(m)}(t)$ represent the amount of type-m data that can be delivered over the link from node a to node b on slot t, and let $D_a^{(m)}(t)$ represent the amount of type m-data that can be dropped from node a on slot t. Note that $Q_n^{(m)}(t) = 0$ for all t if $n = dest(m)$, because once data reaches its destination it is removed from the network. The queueing dynamics are given for all sessions m and all nodes $n \neq dest(m)$ by:

$$\begin{aligned}Q_n^{(m)}(t+1) \quad &\leq \quad \max[Q_n^{(m)}(t) - \sum_{b=1}^{N}\mu_{nb}^{(m)}(t) - D_n^{(m)}(t), 0] \\ &+ \sum_{a=1}^{N}\mu_{an}^{(m)}(t) + 1_n^{(m)} R_m(t)\end{aligned} \quad (27)$$

where $1_n^{(m)}$ is an indicator function that is 1 if $n = source(m)$, and 0 else. The above is an inequality because the *actual* amount of endogenous arrivals $\sum_{a=1}^{N}\tilde{\mu}_{an}^{(m)}(t)$ may be less than or equal to $\sum_{a=1}^{N}\mu_{an}^{(m)}(t)$.

Let $\mathcal{L}^{(m)}$ be the set of all *acceptable directed links* that can be used by data of type m. We assume this set limits the hop-count of all session-m paths to the destination by some maximum number of hops $H^{(m),max}$. If we can limit the delay at each node by some value $W^{(m),max}$, then the total end-to-end delay of all non-dropped data of type-m is at most $H^{(m),max}W^{(m),max}$.

The rates $\mu_{ab}^{(m)}(t)$ are chosen as non-negative values every slot t subject to the following constraints:

$$\textstyle\sum_{m=1}^{M}\mu_{ab}^{(m)}(t) \leq \hat{\mu}_{ab}(\alpha(t), S(t))$$

where $\alpha(t) \in \mathcal{A}_{S(t)}$. The $S(t)$ value again represents channel conditions on each link for slot t, and can depend on the mobility of the network. We assume the sum transmission rates into and out of a node n on any slot are bounded by finite constants $\mu_n^{max,out}$, and $\mu_n^{max,in}$. The $D_n^{(m)}(t)$ decisions at each node are chosen subject to:

$$0 \leq D_n^{(m)}(t) \leq D_n^{(m),max}$$

for some finite constants $D_n^{(m),max}$.

A. Virtual Queues for Multi-Hop

Let $Z_n^{(m)}(t)$ be an ϵ-persistent service virtual queue for type-m data at node n, with dynamics (compare with (3)):

$$\begin{aligned} &Z_n^{(m)}(t+1) = \\ &\quad \max[Z_n^{(m)}(t) + 1_{\{Q_n^{(m)}(t)>0\}}(\epsilon - \textstyle\sum_{b=1}^N \mu_{nb}^{(m)}(t)) \\ &\quad -D_n^{(m)}(t) - 1_{\{Q_m(t)=0\}}\mu_n^{max,out}, 0] \end{aligned} \quad (28)$$

for some value $\epsilon > 0$ (for simplicity, here we consider the same ϵ used at each node).

As before, for each source m we use auxiliary variables $\gamma_m(t)$, chosen in the interval $0 \leq \gamma_m(t) \leq A_m^{max}$, and use the same flow state queue $Y_m(t)$ given in (15).

B. Multi-Hop Drift

Define $\boldsymbol{\Theta}(t) \triangleq [(Q_n^{(m)}(t)), (Z_n^{(m)}(t)), (Y_m(t))]$ as the collection of all virtual and actual queues, and define the Lyapunov function:

$$L(\boldsymbol{\Theta}(t)) \triangleq \frac{1}{2}\sum_{n,m}[Q_n^{(m)}(t)^2 + Z_n^{(m)}(t)^2] + \frac{1}{2}\sum_{m=1}^M Y_m(t)^2$$

Analogous to Lemma 2, the 1-slot drift-plus-penalty can be shown to satisfy:

$$\begin{aligned} \Delta_1(\boldsymbol{\Theta}(t)) - V\sum_{m=1}^M g_m(\gamma_m(t)) + V\sum_{n,m}\beta\nu_m D_n^{(m)}(t) \leq \\ C - \sum_{n,m} Z_n^{(m)}(t) 1_{\{Q_n^{(m)}(t)=0\}}\mu_n^{max,out} \\ -V\sum_{m=1}^M g_m(\gamma_m(t)) + V\sum_{n,m}\beta\nu_m D_n^{(m)}(t) \\ + \sum_{n,m} Q_n^{(m)}(t)\left[1_n^{(m)} R_m(t) + \sum_{a=1}^N \mu_{an}^{(m)}(t)\right] \\ - \sum_{n,m} Q_n^{(m)}(t)[\sum_{b=1}^N \mu_{nb}^{(m)}(t) + D_n^{(m)}(t)] \\ + \sum_{n,m} Z_n^{(m)}(t) 1_{\{Q_n^{(m)}(t)>0\}}[\epsilon - \sum_{b=1}^N \mu_{nb}^{(m)}(t)] \\ - \sum_{n,m} Z_n^{(m)}(t) D_n^{(m)}(t) \\ + \sum_{m=1}^M Y_m(t)[\gamma_m(t) - R_m(t)] \end{aligned}$$

where C is a finite constant that depends on N, M, ϵ, A_m^{max}, $D_n^{(m),max}$, and the $\mu_n^{max,in}$ and $\mu_n^{max,out}$ constants.

The following algorithm is designed to make decisions that minimize the right-hand-side in the above drift bound: Every slot t, observe the current $S(t)$ and the current queue backlogs $(Q_n^{(m)}(t))$, $(Z_n^{(m)}(t))$, $(Y_m(t))$, and make decisions as follows:

- (Resource Allocation) Choose $\alpha(t) \in \mathcal{A}_{S(t)}$ in an effort to maximize the following expression:[4]

$$\textstyle\sum_{a=1}^N \sum_{b=1}^N J_{ab}(t)\hat{\mu}_{ab}(\alpha(t), S(t)) \quad (29)$$

  where $J_{ab}(t)$ is a *differential backlog metric* defined as follows:

$$\begin{aligned} J_{ab}^{(m)}(t) &\triangleq [Q_a^{(m)}(t) + Z_a^{(m)}(t) 1_{\{Q_a^{(m)}(t)>0\}} \\ &\quad -Q_b^{(m)}(t)] 1_{ab}^{(m)} \\ J_{ab}(t) &\triangleq \max_{m\in\{1,\ldots,M\}} \max[0, J_{ab}^{(m)}(t)] \end{aligned}$$

  where $1_{ab}^{(m)}$ is an indicator function that is 1 if link (a,b) can be used by data of type m, that is if $(a,b) \in \mathcal{L}^{(m)}$, and is 0 else.
- (Routing) Define $m_{ab}^*(t) \triangleq \arg\max_{\{m|(a,b)\in\mathcal{L}^{(m)}\}} J_{ab}^{(m)}(t)$. Choose $(\mu_{ab}^{(m)}(t))$ for each link (a,b) as follows:

$$\mu_{ab}^{(m)}(t) = \begin{cases} \hat{\mu}_{ab}(\alpha(t), S(t)) & \text{if } m = m_{ab}^*(t) \\ & \text{ and } J_{ab}^{(m)}(t) \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

- (Dropping) For each node n and data type m, choose $D_n^{(m)}(t)$ by:

$$\begin{aligned} &D_n^{(m)}(t) \\ &= \begin{cases} D_n^{(m),max} & \text{if } Z_n^{(m)}(t) + Q_n^{(m)}(t) > V\beta\nu_m \\ 0 & \text{otherwise} \end{cases} \end{aligned} \quad (30)$$

- (Admission) For each session m: If $Q_{source(m)}^{(m)}(t) \leq Y_m(t)$, then choose $R_m(t) = A_m(t)$. Else, choose $R_m(t) = 0$.
- (Auxiliary Variables) For each session m, choose $\gamma_m(t)$ according to the following optimization:

$$\begin{aligned} \text{Maximize:} \quad & Vg_m(\gamma_m(t)) - Y_m(t)\gamma_m(t) \\ \text{Subject to:} \quad & 0 \leq \gamma_m(t) \leq A_m^{max} \end{aligned}$$

- (Queue Updates) Update queues via (27), (28), (15), using $D_n^{(m)}(t)$ in these updates.

Note that the auxiliary variable and admission decisions $\gamma_m(t)$, $R_m(t)$ are distributed over each source, and the packet drops $D_n^{(m)}(t)$ are distributed over each node. The main difficulty is the max-weight transmission scheduling (29), and Theorem 2 considers constant factor approximations of this.

C. The Multi-Hop T-Slot Lookahead Problem

Define $\mathcal{D}$ as the set of all (n,m) such that type m data can be in node n, so that $Q_n^{(m)}(t) = 0$ for all t whenever $(n,m) \notin \mathcal{D}$. Fix integers $K > 0$ and $T > 0$, and again consider the first KT slots decomposed into K frames of size T. For $k \in \{0, \ldots, K-1\}$, the value $util_k^*(T,\epsilon)$ is defined as

[4]This resource allocation maximization can be difficult in networks with interference, and later we consider constant-factor approximations.

the supremum utility associated with the following problem for frame k:

$$\text{Max: } \sum_{m=1}^{M} g_m\left(\frac{1}{T}\sum_{\tau=kT}^{kT+T-1} R_m(\tau)\right) - V\sum_{n,m}\beta\nu_m d_n^{(m)}$$

Subject to:

$$\sum_{\tau=kT}^{kT+T-1}[1_n^{(m)}R_m(\tau) + \sum_{a=1}^{N}\mu_{an}^{(m)}(\tau)]$$
$$-\sum_{\tau=kT}^{kT+T-1}[\sum_{b=1}^{N}\mu_{nb}^{(m)}(\tau) + d_n^{(m)}] \leq 0 \quad \forall (n,m)\in\mathcal{D}$$
$$\sum_{\tau=kT}^{kT+T-1}[\sum_{b=1}^{N}\mu_{nb}^{(m)}(\tau) + d_n^{(m)}] \geq \epsilon \quad \forall (n,m)\in\mathcal{D}$$
$$0 \leq d_n^{(m)} \leq D_n^{(m),max} \quad \forall n,m,\tau$$
$$\mu_{ab}^{(m)}(\tau) = 0 \text{ if } (a,b)\notin\mathcal{L}^{(m)} \quad \forall (a,b),m,\tau$$
$$0 \leq \mu_{ab}^{(m)}(\tau) \ , \ \sum_{m=1}^{M}\mu_{ab}^{(m)}(\tau) \leq \mu_{ab}(\tau) \quad \forall (a,b),m,\tau$$
$$\boldsymbol{\mu}(\tau)\triangleq(\mu_{ab}(\tau)) \in \mathcal{B}(S(\tau)) \quad \forall \tau$$

where $\mathcal{B}(S(\tau))$ is the convex hull of all values $(\hat{\mu}_{ab}(\alpha, S(\tau)))_{a,b=1}^N$ that can be achieved over decisions $\alpha \in S(\tau)$.

Assuming that $\epsilon \leq D_n^{(m),max}$, it is easy to show the above constraints are always feasible (consider the trivial strategy $\mu_{ab}^{(m)}(\tau) = R_m(\tau) = 0$, $d_n^{(m)} = \epsilon$).

D. Bounded Queues

Recall that $1_n^{(m)}$ is an indicator that is 1 if node n is the source of flow m traffic, and 0 else. We assume that:

$$D_n^{(m),max} \geq \max[\epsilon, 1_n^{(m)}A_m^{max} + \mu_n^{max,in}] \qquad (31)$$

Lemma 5: Assume that (31) holds. Then under the above algorithm for multi-hop routing, dropping, auxiliary variable allocation, and queue updates (possibly using a choice of $\alpha(t)$ other than the max-weight decision of (29)), we have that all queues are bounded for all $t \geq 0$ as follows:

$$Q_n^{(m)}(t) \leq Q_n^{(m),max}, \ Z_n^{(m)}(t) \leq Z_n^{(m),max}, \ Y_m(t) \leq Y_m^{max}$$

provided that these inequalities hold at $t = 0$. The queue bounds are given by:

$$Q_n^{(m),max} \triangleq V\beta\nu_m + \mu_n^{max,in} + 1_n^{(m)}A_m^{max} \qquad (32)$$
$$Z_n^{(m),max} \triangleq V\beta\nu_m + \epsilon \qquad (33)$$
$$Y_m^{max} \triangleq V\nu_m + A_m^{max} \qquad (34)$$

Thus, the worst-case delay of type-m data in a node n is given by an $O(V)$ constant $W_n^{(m),max}$, specifically defined by:

$$W_n^{(m),max}\triangleq\lceil(Q_n^{(m),max} + Z_n^{(m),max})/\epsilon\rceil \qquad (35)$$

Proof: The proof that $Y_m(t) \leq Y_m^{max}$ for all t is the same as that given in Lemma 3. To prove the $Q_n^{(m)}(t)$ bound, suppose that $Q_n^{(m)}(t) \leq Q_n^{(m),max}$ at some slot t. We show it also holds for slot $t+1$. If $Q_n^{(m)}(t) \leq V\beta\nu_m$, then $Q_n^{(m)}(t+1) \leq V\beta\nu_m + \mu_n^{max,in} + 1_n^{(m)}A_m^{max}\triangleq Q_n^{(m),max}$. Else, we have $Q_n^{(m)}(t) > V\beta\nu_m$, and so the dropping algorithm (30) yields $D_n^{(m)}(t) = D_n^{(m),max}$. By (31), $D_n^{(m),max}$ is greater than or equal to the maximum new arrivals of type m on slot t, and so we again have $Q_n^{(m)}(t+1) \leq Q^{(m),max}$. The proof of the $Z_n^{(m),max}$ bound is similar and omitted. The worst-case delay result (35) then follows by Lemma 1. □

E. Multi-Hop Utility Performance

It is important to note that the queue backlog and delay bounds (32)-(35) hold for *any* resource allocation decisions $\alpha(t)$. To analyze utility, assume the above dynamic algorithm is used, with the exception that the resource allocation can come within a factor θ of the max-weight decision (29) every slot (where $0 < \theta \leq 1$), called θ-max-weight. The following theorem has a proof that is similar to the 1-hop case:

Theorem 2: Assume that (31) holds, that all virtual and actual queues are initially empty, and that the above dynamic algorithm (with θ-max-weight) is used. Assume that for all frames $k \in \{0, 1, \ldots, K-1\}$, the constraints in the multi-hop T-slot lookahead problem are feasible when the fourth constraint involving ϵ is changed to $\epsilon' = \epsilon/\theta$ (this feasibility holds whenever $\epsilon' \leq D_n^{(m),max}$ for all n, m). Then for all integers $K > 0$, $T > 0$, the throughput-utility over the first KT slots satisfies:

$$\begin{aligned}\textstyle\sum_{m=1}^M g_m(\overline{r}_m(KT)) - \sum_{n,m}\beta\nu_m\overline{d}_n^{(m)}(kT) \geq \\ \textstyle\frac{1}{K}\sum_{k=0}^{K=1}\sum_{m=1}^M g_m(\theta r_m^*(k,T,\epsilon/\theta)) \\ \textstyle-\theta\sum_{n,m}\beta\nu_m d_n^{(m)*}(k,T,\epsilon/\theta) \\ \textstyle-\frac{CT}{V} - \frac{1}{KT}\sum_{m=1}^M \nu_m Y_m^{max}\end{aligned}$$

where $\overline{r}_m(KT)$, $\overline{d}_n^{(m)}(kT)$ are time averages of $R_m(t)$ and $D_n^{(m)}(t)$ over the first KT slots, the constant C is independent of T and V, and where $r_m^*(k,T,\epsilon/\theta)$, $d_n^{(m)*}(k,T,\epsilon/\theta)$ are optimal values associated with the T-slot lookahead problem for frame k, frame size T, and parameter $\epsilon' = \epsilon/\theta$.

We note that if $\theta = 1$ (so pure max-weight is used), then the first two terms on the right-hand-side above are equal to:

$$\textstyle\frac{1}{K}\sum_{k=0}^{K-1} util_k^*(T,\epsilon)$$

The above theorem shows that, as $K \to \infty$, the achieved throughput-utility is within $O(1/V)$ of the θ-scaled target.

We recall that if the allowed routing paths $\mathcal{L}^{(m)}$ have at most $H^{(m),max}$ hops, then the worst case delay at each node translates into an end-to-end worst case delay of $H^{(m),max}W^{(m),max}$. One can likely avoid specifying hop-count-limited paths by combining the sample-path and ϵ-persistent queue techniques developed here with the hop-count based queue architecture in [19].

IV. Conclusions

We have used a new ϵ-persistent virtual queue technique together with a sample path analysis to develop utility-efficient scheduling algorithms that provide worst-case delay guarantees. Our algorithms yield, for arbitrary sample paths, throughput-utility that is within $O(1/V)$ of the target, which can be made arbitrarily small with a worst-case delay tradeoff that is $O(V)$. These techniques can also be used in other scheduling contexts where worst-case delay is important.

V. APPENDIX A — PROOF OF LEMMA 2

We use the fact that $(\max[Q-b,0]+a)^2 \leq Q^2+a^2+b^2+2Q(a-b)$ for any $Q \geq 0$, $b \geq 0$, $a \geq 0$. Squaring the update for $Q_m(t)$ in (1) and using $R_m(t)^2 \leq (A_m^{max})^2$, $(\mu_m(t)+D_m(t))^2 \leq (\mu_m^{max}+D_m^{max})^2$ gives:

$$\begin{aligned} Q_m(t+1)^2 - Q_m(t)^2 \leq (A_m^{max})^2 + (\mu_m^{max}+D_m^{max})^2 \\ +2Q_m(t)(R_m(t)-\mu_m(t)-D_m(t)) \end{aligned}$$

Squaring the update for $Y_m(t)$ in (15), using the fact that $\max[x,0]^2 \leq x^2$ and $(R_m(t)-\gamma_m(t))^2 \leq (A_m^{max})^2$ gives:

$$Y_m(t+1)^2 - Y_m(t)^2 \leq (A_m^{max})^2 + 2Y_m(t)(\gamma_m(t)-R_m(t))$$

For simplicity of notation, define $I_m(t)\triangleq 1_{\{Q_m(t)>0\}}$ and $J_m(t)\triangleq 1_{\{Q_m(t)=0\}}$. Squaring the update for $Z_m(t)$ in (3), using the fact that $\max[x,0]^2 \leq x^2$, gives:

$$\begin{aligned} Z_m(t+1)^2 \leq Z_m(t)^2 \\ +[I_m(t)(\epsilon_m-\mu_m(t)) - D_m(t) - J_m(t)\mu_m^{max}]^2 \\ +2Z_m(t)[I_m(t)(\epsilon_m-\mu_m(t)) - D_m(t) - J_m(t)\mu_m^{max}] \end{aligned} \quad (36)$$

We now have the following observation:

$$\begin{aligned} [I_m(t)(\epsilon_m-\mu_m(t)) - D_m(t) - J_m(t)\mu_m^{max}]^2 \leq \\ \max[\epsilon_m^2, (\mu_m^{max}+D_m^{max})^2] \end{aligned} \quad (37)$$

Inequality (37) holds by the following argument: If $I_m(t)=1$, then $J_m(t)=0$, and we have:

$$\begin{aligned} [I_m(t)(\epsilon_m-\mu_m(t)) - D_m(t) - J_m(t)\mu_m^{max}]^2 = \\ (\epsilon_m-\mu_m(t)-D_m(t))^2 \leq \\ \max[\epsilon_m^2, (\mu_m^{max}+D_m^{max})^2] \end{aligned}$$

In the opposite case when $I_m(t)=0$, then $J_m(t)=1$ and:

$$\begin{aligned} &[I_m(t)(\epsilon_m-\mu_m(t)) - D_m(t) - J_m(t)\mu_m^{max}]^2 \\ &\quad \leq \quad (D_m(t)+\mu_m^{max})^2 \\ &\quad \leq \quad \max[\epsilon_m^2, (\mu_m^{max}+D_m^{max})^2] \end{aligned}$$

Thus, in all cases, (37) holds.

Substituting (37) into (36) yields:

$$\begin{aligned} Z_m(t+1)^2 - Z_m(t)^2 \leq \max[\epsilon_m^2, (\mu_m^{max}+D_m^{max})^2] \\ +2Z_m(t)[I_m(t)(\epsilon_m-\mu_m(t)) - D_m(t) - J_m(t)\mu_m^{max}] \end{aligned}$$

Summing the squared differences in the queues over all m and dividing by 2 yields the result of Lemma 2.

REFERENCES

[1] 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.

[2] M. J. Neely. Delay-based network utility maximization. *Proc. IEEE INFOCOM*, March 2010.

[3] F.P. Kelly, A.Maulloo, and D. Tan. Rate control for communication networks: Shadow prices, proportional fairness, and stability. *Journ. of the Operational Res. Society*, vol. 49, no. 3, pp. 237-252, March 1998.

[4] S. H. Low and D. E. Lapsley. Optimization flow control, i: Basic algorithm and convergence. *IEEE/ACM Transactions on Networking*, vol. 7 no. 6, pp. 861-875, Dec. 1999.

[5] M. Chiang. Balancing transport and physical layer in wireless multihop networks: Jointly optimal congestion control and power control. *IEEE J. on Selected Areas in Comm.*, vol. 23, no. 1, pp. 104-116, Jan. 2005.

[6] M. Chiang, S. H. Low, A. R. Calderbank, and J. C. Doyle. Layering as optimization decomposition: A mathematical theory of network architectures. *Proceedings of the IEEE*, vol. 95, no. 1, Jan. 2007.

[7] X. Liu, E. K. P. Chong, and N. B. Shroff. A framework for opportunistic scheduling in wireless networks. *Computer Networks*, vol. 41, no. 4, pp. 451-474, March 2003.

[8] R. Agrawal and V. Subramanian. Optimality of certain channel aware scheduling policies. *Proc. 40th Annual Allerton Conference on Communication , Control, and Computing, Monticello, IL*, Oct. 2002.

[9] H. Kushner and P. Whiting. Asymptotic properties of proportional-fair sharing algorithms. *Proc. of 40th Annual Allerton Conf. on Communication, Control, and Computing*, 2002.

[10] M. J. Neely. *Dynamic Power Allocation and Routing for Satellite and Wireless Networks with Time Varying Channels*. PhD thesis, Massachusetts Institute of Technology, LIDS, 2003.

[11] M. J. Neely, E. Modiano, and C. Li. Fairness and optimal stochastic control for heterogeneous networks. *Proc. IEEE INFOCOM*, March 2005.

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

[13] L. Tassiulas and A. Ephremides. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. *IEEE Transacations on Automatic Control*, vol. 37, no. 12, pp. 1936-1948, Dec. 1992.

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

[15] E. Leonardi, M. Mellia, F. Neri, and M. Ajmone Marsan. Bounds on average delays and queue size averages and variances in input-queued cell-based switches. *Proc. IEEE INFOCOM*, 2001.

[16] N. McKeown, A. Mekkittikul, V. Anantharam, and J. Walrand. Achieving 100% throughput in an input-queued switch. *IEEE Transactions on Communications*, vol. 47, no. 8, August 1999.

[17] P. R. Kumar and S. P. Meyn. Stability of queueing networks and scheduling policies. *IEEE Trans. on Automatic Control*, vol.40,.n.2, pp.251-260, Feb. 1995.

[18] L. B. Le, E. Modiano, and N. B. Shroff. Optimal control of wireless networks with finite buffers. *Proc. IEEE INFOCOM*, 2010.

[19] L. Ying, S. Shakkottai, and A. Reddy. On combining shortest-path and back-pressure routing over multihop wireless networks. *Proc. IEEE INFOCOM*, 2009.

[20] A. Eryilmaz and R. Srikant. Fair resource allocation in wireless networks using queue-length-based scheduling and congestion control. *Proc. IEEE INFOCOM*, March 2005.

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

[22] X. Lin and N. B. Shroff. The impact of imperfect scheduling on cross-layer rate control in wireless networks. *Proc. IEEE INFOCOM*, 2005.

[23] X. Lin and N. B. Shroff. Joint rate control and scheduling in multihop wireless networks. *Proc. of 43rd IEEE Conf. on Decision and Control, Paradise Island, Bahamas*, Dec. 2004.

[24] B. Sadiq, S. Baek, and Gustavo de Veciana. Delay-optimal opportunistic scheduling and approximations: the log rule. *Proc. IEEE INFOCOM*, April 2009.

[25] A. Wierman, L. L. H. Andrew, and A. Tang. Power-aware speed scaling in processor sharing systems. *Proc. IEEE INFOCOM*, Rio de Janiero, Brazil, April 2009.

[26] A. Fu, E. Modiano, and J. Tsitsiklis. Optimal energy allocation for delay-constrained data transmission over a time-varying channel. *Proc. IEEE INFOCOM*, 2003.

[27] E. Uysal-Biyikoglu, B. Prabhakar, and A. El Gamal. Energy-efficient packet transmission over a wireless link. *IEEE/ACM Trans. Networking*, vol. 10, no. 4, pp. 487-499, Aug. 2002.

[28] M. Zafer and E. Modiano. A calculus approach to minimum energy transmission policies with quality of service guarantees. *Proc. IEEE INFOCOM*, March 2005.

[29] W. Chen, M. J. Neely, and U. Mitra. Energy-efficient transmissions with individual packet delay constraints. *IEEE Transactions on Information Theory*, vol. 54, no. 5, pp. 2090-2109, May 2008.

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