Low-Complexity Scheduling Algorithm for Sum-Queue Minimization in Wireless Convergecast

V.J. Venkataramanan and Xiaojun Lin
School of ECE, Purdue University
Email: {vvenkat,linx}@purdue.edu

Abstract

We consider the problem of link scheduling for efficient convergecast in a wireless system. While there have been many results on scheduling algorithms that attain the maximum possible throughput in such a system, there has been few results that provide scheduling algorithms that are optimal in terms of some quality-of-service metric such as the probability that the end-to-end buffer usage exceeds a large threshold. Using a large deviations framework, we design a novel and low complexity algorithm that attains the best asymptotic decay rate of the probability of sum-queue overflow as the overflow threshold becomes large. Through simulations, we show that this algorithm has better performance than well known algorithms such as the standard back-pressure algorithm and the multihop version of GMM (combined with back-pressure). Our algorithm performs better not only in terms of the asymptotic decay rate at large overflow thresholds, but also in terms of the actual probability of overflow for practical range of overflow thresholds.

I. Introduction

We consider the link scheduling problem in a wireless multihop network for convergecast. In a convergecast situation, there is a central node to which each node in the network forwards data. We assume that each node uses a fixed path to route data to the central node. The same path is used irrespective of whether a node is relaying data for other nodes or is transmitting its own data. This scenario results in a tree topology for the overall system of flows. Such convergecast problems arise commonly in sensor networks where there is a central node that collects data, such as temperature readings or audio/video signals, from all other nodes in the network.

Since the communication medium is wireless, managing interference becomes important in order to achieve effective data transfer. For example, allowing all nodes to transmit at the same time would lead to a significant number of collisions whereas allowing only one node to talk at a time will be inefficient. Scheduling algorithms play the crucial role of determining which links in the system can be activated at a certain time. A first order requirement made on any desirable scheduling algorithm is that of throughput optimality. That is, if the desired scheduling algorithm can not stabilize the system for an offered load, then no other algorithm can stabilize the system for the same offered load. There has been a substantial amount of work in this area of research starting with the seminal work of Tassiulas and Ephremides [1]. The backpressure algorithm [1], [2], exponential rule [3], log rule [4], [5] and α-algorithms [6] are several throughput optimal algorithms known to date.

However, throughput-optimality alone is not sufficient when performance metrics such as mean-delay, probability of delay violation or probability of buffer overflow are considered. For example, the well known throughput-optimal back pressure algorithm suffers from large delays [7], [8]. Many throughput-optimal algorithms make their scheduling decisions based on the backlog in the system, which in turn depends on past scheduling decisions and arrival rates. Such cross-dependency results in

system dynamics that are difficult to analyze. Due to this reason, the behavior of throughput-optimal algorithms in terms of finer QoS performance metrics, such as those mentioned before, is difficult to quantify. For *single-hop* traffic several techniques that have been used in the past are mean-delay analysis [9], [10], heavy traffic analysis [3], [11] and large deviations [6], [12]–[15]. For *multi-hop* traffic, however the problem becomes even more complex due to the coupling between the departure process of a node and the arrival process of the node downstream. There are few results in this case. [16], [17] study mean-delay performance in the presence of multi-hop traffic. While [16] provides lower bounds on the mean-delay, it does not immediately reveal which algorithm is optimal. [17] provides an algorithm that is order-optimal for mean-delay. While the algorithm achieves the optimal order when the number of nodes is large, for small or moderate size systems, the algorithm may not be close to optimal. [18] studies a specific tandem network topology and proves sample path optimality of the sum-queue.

Instead of analyzing the mean-delay performance as in [16], [17], we seek to design a scheduling algorithm to minimize the probability that the sum-queue of the system exceeds a large threshold. However, it appears difficult to apply the technique of [18] in this more general network topology and to derive the exact probability for sum-queue overflow. Hence, we employ a large deviations approximation which captures the asymptotic rate of decay of this probability as the threshold increases [13].

We design a low complexity scheduling algorithm called P-TREE algorithm that attains the maximum rate of decay for the probability of overflow of the sum backlog. The philosophy behind the algorithm is to ensure that as much data is driven out of the system as soon as possible. The algorithm achieves this by giving priority to links that are closer to the destination node and to links which have a larger capacity. The details of this algorithm are provided in section IV. A non-rigorous explanation of the P-TREE algorithm is the following. The algorithm considers for activation all the links attached to the root, then it considers all the links at depth 2 from the root and so on till it has reached the deepest leaf nodes. At each depth, the algorithm first eliminates from consideration all links that might interfere with links already activated at a lower depth. Then, from the remaining links, the algorithm activates those links which will lead to a maximum net transfer of data across that depth. When applied to the tandem topology studied in [18], our algorithm reduces to the algorithm used in [18]. However, we emphasize that our algorithm works for the more general tree topology.

Simulation results show that the algorithm significantly reduces the probability of sum-queue overflow even when the overflow threshold is not very large. It performs much better than both the backpressure algorithm and the multi-hop version of the low-complexity greedy maximal matching (GMM) algorithm. Take L to be the number of links in the system. For the scenario we consider (1-hop interference), it is known that the back-pressure algorithm has a complexity of $O(L^3)$ [19] and the greedy maximal matching algorithm has a complexity of $O(L \log L)$ [20]. In comparison, the P-Tree algorithm has an even lower complexity of $O(L)$.

The large-deviations optimality of the P-TREE algorithm is based on a result from our earlier work that, under suitable assumptions, an algorithm that minimizes the drift of a Lyapunov function at every time in every fluid sample path is large-deviations optimal for minimizing the probability that the Lyapunov function overflows [13]. However, as we will discuss later, it is not trivial to come up with the P-Tree algorithm and to verify that it minimizes the drift of the sum-queue in every fluid sample paths. As readers will see in Section V, such verification involves novel techniques that uncover non-trivial insights on the dynamics of the P-Tree algorithm. These techniques are of independent interest and may be useful for other settings as well.

The rest of the paper is organized as follows. Section II describes the system model and the performance objective. Section

IV describes the P-TREE scheduling algorithm and in section V, we carry out the theoretical analysis to show optimality of the priority algorithm. Section VI provides simulation results.

II. System Model

As mentioned in the introduction, the convergecast problem leads to a tree topology for the flows in the network. The root of the tree is the destination node for all flows in the network. Each flow in the network originates at some node of the tree (other than the root) and follows the shortest path to the root. There can be only one flow originating at each node. All nodes in the tree other than the root and leaf nodes will be referred to as interior nodes. Each node (except the root) in the network can be associated with a link that connects the node to its parent node. Since this association is a one-to-one mapping, we will use a unique identifier $\boldsymbol{l}$ to refer to both the link and its corresponding node. We will use $C(\boldsymbol{l})$ to denote the number of children of node $\boldsymbol{l}$ and C to denote the number of children of the root. Let $\mathcal{L}$ denote the set of all links/nodes in the network.

For ease of exposition, we use a vector $\boldsymbol{l}$ to identify a link (node), which can be explained through the following recursive procedure. Consider that we have labeled a node as $\boldsymbol{l} = (l_1, \ldots, l_{D(\boldsymbol{l})})$ (with $D(\boldsymbol{l})$ denoting the dimension of $\boldsymbol{l}$). Then, the task of labeling its child nodes is accomplished as follows. The child nodes are ordered according to their link capacities. The one with the highest link capacity is labelled $(l_1, \ldots, l_{D(\boldsymbol{l})}, 1)$, the next is labelled $(l_1, \ldots, l_{D(\boldsymbol{l})}, 2)$ and so on. In the future, we will use the notation $< \boldsymbol{l}, i >$ to denote the vector $(l_1, \ldots, l_{D(\boldsymbol{l})}, i)$ and $< \boldsymbol{l}, i, j >$ to denote $<< \boldsymbol{l}, i >, j >$. To start off this procedure, we label the root node with a null vector. Hence the vectors of dimension one, i.e $1, 2, \ldots, C$ represent the nodes at depth 1 arranged in decreasing order of their link capacities. Please see fig 1 for an example of the labeling scheme.

The interference model we consider is the so-called one-hop interference model [20], [21]. This means that a node can either receive or transmit during a time-slot but not both. Further, it can only receive from one of its children nodes at a time. This interference model is useful for Bluetooth, UWB, FH-CDMA systems [20], [22], [23]. We assume that time is slotted and the link capacity is fixed at all time. Let $F_{\boldsymbol{l}}$ denote the capacity of link $\boldsymbol{l}$, i.e. the amount of data that can be transmitted over link $\boldsymbol{l}$ in a time-slot is $F_{\boldsymbol{l}}$ provided interfering links are silent.

The queue associated with link $\boldsymbol{l}$, denoted by $X_{\boldsymbol{l}}(t)$, is maintained by node $\boldsymbol{l}$. Let $E_{\boldsymbol{l}}(t)$ denote the amount of data transmitted over link $\boldsymbol{l}$ in time-slot t. We impose the constraint that $E_{\boldsymbol{l}}(t) < X_{\boldsymbol{l}}(t)$. Let $A_{\boldsymbol{l}}(t)$ denote the amount of data generated by node $\boldsymbol{l}$ in time-slot t. We assume that $A_{\boldsymbol{l}}(t)$ is i.i.d in time* and that there is a bound M on the maximum amount of data that any node can generate in a time-slot. Let $\hat{\lambda}_{\boldsymbol{l}} \triangleq E(A_{\boldsymbol{l}}(t))$ be the expected arrival rate. We assume that $\hat{\boldsymbol{\lambda}}$ is such that the system is stabilizable, i.e. there exists some scheduling algorithm that can stabilize the system. The queue evolution is then as follows

$$\begin{aligned} X_{\boldsymbol{l}}(t+1) &= X_{\boldsymbol{l}}(t) + A_{\boldsymbol{l}}(t) + \sum_{l=1}^{C(\boldsymbol{l})} E_{<\boldsymbol{l},l>}(t) - E_{\boldsymbol{l}}(t) \\ &\quad \text{if } \boldsymbol{l} \text{ is not a leaf} \\ X_{\boldsymbol{l}}(t+1) &= X_{\boldsymbol{l}}(t) + A_{\boldsymbol{l}}(t) - E_{\boldsymbol{l}}(t) \text{ if } \boldsymbol{l} \text{ is a leaf} \end{aligned} \tag{1}$$

Note that the root maintains no queue since it is the destination node for all flows. Further, note that $X_{\boldsymbol{l}}(t) \geq 0$ for all t and links $\boldsymbol{l}$. From (1), we can derive

$$\sum_{\boldsymbol{l} \in \mathcal{L}} X_{\boldsymbol{l}}(t+1) = \sum_{\boldsymbol{l} \in \mathcal{L}} X_{\boldsymbol{l}}(t) + \sum_{\boldsymbol{l} \in \mathcal{L}} A_{\boldsymbol{l}}(t) - \sum_{l=1}^{C} E_l(t). \tag{2}$$

*This assumption can be relaxed. It suffices that $A_{\boldsymbol{l}}(t)$ satisfy a sample path LDP [13].

(2) says that the sum queue is governed by a simple queueing equation where the arrival is the sum of the arrivals at each node in the tree and the service is the sum of service given to the links connected to the root. Note that the service given to any other link in the system will not change the sum queue since it is simply an internal transfer of data.

A. Performance Objective

In this paper, we are interested in designing a scheduling algorithm to minimize the total buffer occupancy in the network in the following sense. We want to minimize the steady-state probability that the total buffer occupancy exceeds a threshold B. The precise mathematical quantity that we want to minimize is given by

$$\mathbf{P}\left[\sum_{\boldsymbol{l}\in\mathcal{L}} X_{\boldsymbol{l}}(0) \geq B\right]. \tag{3}$$

Unfortunately, in general this quantity is mathematically intractable. We instead use the following large deviations quantities

$$-I \triangleq \liminf_{B\to\infty} \frac{1}{B}\log\left(\mathbf{P}\left[\sum_{\boldsymbol{l}\in\mathcal{L}} X_{\boldsymbol{l}}(0) \geq B\right]\right) \tag{4}$$

$$-J \triangleq \limsup_{B\to\infty} \frac{1}{B}\log\left(\mathbf{P}\left[\sum_{\boldsymbol{l}\in\mathcal{L}} X_{\boldsymbol{l}}(0) \geq B\right]\right) \tag{5}$$

to provide an approximation of (3). Note that for large B, we have

$$e^{-IB+o(B)} \leq \mathbf{P}\left[\sum_{\boldsymbol{l}\in\mathcal{L}} X_{\boldsymbol{l}}(0) \geq B\right] \leq e^{-JB+o(B)}.$$

The quantities I and J can be determined by the so-called fluid-sample-paths (FSPs) [13] described next.

III. Large Deviations

We first define the concept of fluid sample paths.

A. Fluid Sample Paths

For a fixed B and T, define the following scaled quantities in the time interval $[-T, 0]$:

$$a^B_{\boldsymbol{l}}(t) = \frac{1}{B}\sum_{\tau=0}^{B(T+t)} A_{\boldsymbol{l}}(\tau), \quad x^B_{\boldsymbol{l}}(t) = \frac{1}{B}X_{\boldsymbol{l}}(B(T+t)),$$

$$e^B_{\boldsymbol{l}}(t) = \frac{1}{B}\sum_{\tau=0}^{B(T+t)} E_{\boldsymbol{l}}(\tau). \tag{6}$$

Note that the probabilities in (4) and (5) can now be rewritten as $\mathbf{P}[\sum_{\boldsymbol{l}\in\mathcal{L}} x^B_{\boldsymbol{l}}(t) \geq 1]$. Denote by $\boldsymbol{a}^B(t)$ the vector $[a^B_{\boldsymbol{l}}(t)]_{\boldsymbol{l}\in\mathcal{L}}$. The vectors $\boldsymbol{x}^B(t)$ and $\boldsymbol{e}^B(t)$ are defined similarly. Since the quantities $(\boldsymbol{a}^B(t), \boldsymbol{x}^B(t), \boldsymbol{e}^B(t))$ are Lipschitz continuous, there exists a subsequence over which they converge uniformly over compact intervals (u.o.c). Any such limit is called a fluid-sample-path (FSP). In other words, $(\boldsymbol{a}(t), \boldsymbol{x}(t), \boldsymbol{e}(t))$ is called a FSP if for some $T > 0$ there exists a sequence $(\boldsymbol{a}^B(t), \boldsymbol{x}^B(t), \boldsymbol{e}^B(t))$ that converges to it u.o.c over $[-T, 0]$.

Note that FSPs are different from fluid limits. Fluid limits are limiting processes to which $(\boldsymbol{a}^B(t), \boldsymbol{x}^B(t), \boldsymbol{e}^B(t))$ converge with probability 1. Hence, fluid limits capture the *mean* behavior of the system. In contrast, convergence to an FSP does not need to be with probability 1. Hence an FSP is more general and captures large-deviations behavior that deviates from the mean.

B. Large deviations principle

Since the arrival process is *i.i.d* in time, the sequence of scaled processes $\boldsymbol{a}^B(t)$ satisfies a sample path large deviations principle with some rate function $I_a^T(\cdot)$. What this means is the following. Let $\Phi_a[-T,0]$ be the space of component-wise non-decreasing functions $\boldsymbol{a}(t)$ on $[-T,0]$ with $\boldsymbol{a}(-T)=0$. Let this space be equipped with the essential supremum norm [24, p176]. For any set Γ of trajectories in $\Phi_a[-T,0]$, the probability that the sequence of scaled arrival processes $\boldsymbol{a}^B(t)$ fall into the set Γ satisfies:

$$\lim_{B\to\infty}\frac{1}{B}\log\mathbf{P}[\boldsymbol{a}^B(t)\in\Gamma]=-\inf_{\boldsymbol{a}\in\Gamma}I_a^T(\boldsymbol{a}).$$

That is, the decay-rate of the probability $\mathbf{P}[\boldsymbol{a}^B(t)\in\Gamma]$ is determined by the trajectory $\boldsymbol{a}$ in Γ with the least cost $I_a^T(\boldsymbol{a})$.

Under a given scheduling algorithm, if the mapping from the arrival process $\boldsymbol{a}^B(t)$ to the queue process $\boldsymbol{x}^B(t)$ is continuous, then one can apply the contraction principle [24] and conclude that the sequence of scaled queue processes $\boldsymbol{x}^B(t)$ satisfies a large deviations principle as well. That is, the rate of decay of the probability of overflow $\mathbf{P}[\sum_{\boldsymbol{l}\in\mathcal{L}}\boldsymbol{x}_{\boldsymbol{l}}^B(0)\geq 1]$ (either I in (4) or J in (5)) is determined by the minimum cost $I_a^T(\boldsymbol{a})$ among all trajectories $\boldsymbol{a}(t)$ that causes the queue to grow from $\boldsymbol{a}(-T)=0$ at $t=-T$ to overflow at $t=0$. The trajectory that attains this minimum-cost-to-overflow is called the "most likely path to overflow."

However, for the system that we are interested in, this approach encounters several difficulties. First, for many scheduling algorithms it is very difficult to verify the continuity of the mapping from $\boldsymbol{a}^B(t)$ to $\boldsymbol{x}^B(t)$ and hence the contraction principle may not hold. Second, even if the contraction principle can be applied, it is difficult to find the minimum-cost-to-overflow because we have to solve a multi-dimensional calculus-of-variations problem. Third, even if we can compute the minimum-cost-to-overflow for a given algorithm, it is unclear how to optimize *across algorithms* to find the optimal algorithm.

In our earlier work [13], we establish a new result (Theorem 8 in [13]) that circumvents these difficulties. This result is re-stated in this paper as Theorem 3. Roughly speaking, this Theorem states that under certain assumptions on a Lyapunov function $V(\boldsymbol{x})$, if the scheduling algorithm π_0 minimizes the drift of the Lyapunov function $V(\boldsymbol{x}(t))$ at each point in time in every fluid sample path, then the algorithm π_0 must be large deviations decay-rate optimal for $\mathbf{P}[V(\boldsymbol{x}^B(0))\geq 1]$. This result has the following intuitive explanation: For any given FSP under algorithm π_0, since the algorithm π_0 minimizes the drift of the Lyapunov function at every time, it is plausible that the algorithm π_0 will also minimize the value of the Lyapunov function at the end of the FSP. We note however that this statement is not trivial to hold: minimizing the drift at each point in time is a myopic property, which may not always lead to a globally optimal behavior! In our prior work [13], we have rigorously quantified the conditions under which the above statement holds (see also Theorem 3 in Section V). Once these conditions are satisfied, we can see that for any FSP that leads to overflow under algorithm π_0, the corresponding FSP (with the same arrival process $\boldsymbol{a}(t)$ and thus the same cost $I_a^T(\boldsymbol{a})$) under any other algorithms must also overflow. Hence, the minimum-cost-to-overflow (and thus the decay rate of the overflow probability) under algorithm π_0 must be no smaller than that under any other algorithms.

Hence, our problem becomes that of finding an algorithm (i.e. Algorithm P-Tree in section IV) and verifying that it is drift-minimizing for the Lyapunov function $V(\boldsymbol{x}(t))=\sum_{\boldsymbol{l}\in\mathcal{L}}\boldsymbol{x}_{\boldsymbol{l}}(t)$ at each time in every fluid sample path. This, however, is not a trivial task. Although it is not difficult to identify the minimum drift at a given point in an FSP, it is often much more difficult to find an algorithm *in the original discrete-time system* that attains the minimum drift. This is because drift minimization in FSP, even over an infinitesimally small interval δ, corresponds to the cumulative effect over time interval $B\delta$

in the original discrete-time system. However, in the original discrete-time system, an algorithm cannot know the "future" in the interval $B\delta$ before hand. As a result, it is not always easy to design a discrete-time algorithm that minimzes the drift in each time in every FSP. This discrepancy between discrete-time and fluid-scaled continuous-time was discussed in [25] for fluid limits, where the authors establish conditions under which minimizing the drift in discrete-time is sufficient for minimizing the drift in fluid limits. However, our Lyapunov function $\sum_{\boldsymbol{l}\in\mathcal{L}} X_{\boldsymbol{l}}(t)$ does not satisfy the conditions in [25], and hence the techniques there do not apply. Further, as readers will see, the P-Tree algorithm that we will propose in section IV does not minimize the drift of the Lyapunov function $\sum_{\boldsymbol{l}\in\mathcal{L}} X_{\boldsymbol{l}}(t)$ in discrete-time either. For instance, if a link has insufficient data, i.e $X_{\boldsymbol{l}}(t) < F_{\boldsymbol{l}}$, then that link is not considered for activation in that time-slot even though serving that link might drain more packets from the system. Nonetheless, we will develop new techniques that confirm that the P-Tree algorithm indeed minimizes the drift of the Lyapunov function at each time in every FSP. These techniques reveal non-trivial insights on the dynamics of the system under the P-Tree algorithm.

Finally, we emphasize that our task of proving the drift minimizing property for *fluid sample paths* is distinct from the more common proofs in the literature for proving that certain algorithms are drift minimizing for the *fluid limit*. As stated previously, fluid limits only capture the *mean* behavior where-as FSPs capture behaviors that deviate from the mean as well. Proving an algorithm to be drift minimizing for FSP is inherently more difficult since drift minimization must be shown for any conceivable system behavior.

IV. P-TREE SCHEDULER

We next describe P-TREE, a simple priority based scheduling algorithm tailored for the tree network. The algorithm is based on two guiding principles. First, we give priority to links that are closer to the root (destination) node and secondly, we give priority to links that carry more data per timeslot. The intuition is that by following the two principles, we hope to move data out of the network as fast as possible and hence reduce the total buffer occupancy in the network.

Only links that have enough data to fully utilize the capacity, i.e. $X_{\boldsymbol{l}}(t) > F_{\boldsymbol{l}}$, are considered for activation in time-slot $t^{\dagger}$. Let the set of such links be denoted by $\mathcal{A}(t)$. To choose the link for activation, the algorithm first considers links $\boldsymbol{l}$ of dimension 1. It chooses to activate link $l^* = \min\{l | 1 \leq l \leq C, l \in \mathcal{A}(t)\}$. Recall that the links of dimension 1 are numbered in decreasing order of link capacity. Hence, l^* is the link with the largest capacity among links of dimension 1 that are considered for activation. Then the algorithm considers all interior nodes at depth 1, then all interior nodes at depth 2 and so on till it has considered all interior nodes. Each time the algorithm considers an interior node $\boldsymbol{l}$, it performs the following:

If link $\boldsymbol{l}$ is activated, then none of the links$< \boldsymbol{l}, l >$ ($l = 1, \ldots, C(\boldsymbol{l})$) will be activated (due to interference). Otherwise link $< \boldsymbol{l}, l^* >$ is activated where $l^* = \min\{l | 1 \leq l \leq C(\boldsymbol{l}), < \boldsymbol{l}, l > \in \mathcal{A}(t)\}$. Again, if we recall the structure used to label links, we can see that the above optimization problem is choosing the link with the largest capacity among all contending links.

This algorithm can be considered as a generalization of the algorithm π_0 specified in [18] where the authors establish that for a tandem topology (i.e. a tree with no branching) the algorithm is sample path optimal in terms of the sum queue backlog. In comparison, in this work we show that the P-TREE algorithm is large deviations decay rate optimal in terms of the sum queue backlog for the more general tree topology.

[†]Due to this restriction, the P-Tree algorithm does not always minimize the drift of $\sum_{\boldsymbol{l}\in\mathcal{L}} X_{\boldsymbol{l}}(t)$ in the discrete time.

V. Analysis

Any FSP $(\boldsymbol{a}(t), \boldsymbol{x}(t), \boldsymbol{e}(t))$ is differentiable almost everywhere in the interval $[-T, 0]$ [13]. Denote the set of time instances where the FSP is not differentiable as $\mathcal{T}$. Then $\mathcal{T}$ is of measure 0. In the rest of this paper, we will restrict discussion to time $t \notin \mathcal{T}$, and we will call such time a regular time. Define the following related quantities $f_{\boldsymbol{l}}(t) = \frac{1}{F_{\boldsymbol{l}}}\frac{d}{dt}a_{\boldsymbol{l}}(t)$ and $\mu_{\boldsymbol{l}}(t) = \frac{1}{F_{\boldsymbol{l}}}\frac{d}{dt}e_{\boldsymbol{l}}(t)$.

We remind the readers that $F_{\boldsymbol{l}}f_{\boldsymbol{l}}(t)$ is different from the mean arrival rate $\hat{\lambda}_{\boldsymbol{l}}(t)$ since we are considering an FSP. From the queue evolution equation (1) and (2), we can derive the following for the FSP

$$\begin{aligned}
\frac{d}{dt}x_{\boldsymbol{l}}(t) &= F_{\boldsymbol{l}}f_{\boldsymbol{l}}(t) + \sum_{l=1}^{C(\boldsymbol{l})} F_{<\boldsymbol{l},l>}\mu_{<\boldsymbol{l},l>}(t) - F_{\boldsymbol{l}}\mu_{\boldsymbol{l}}(t), \\
&\quad \text{if } \boldsymbol{l} \text{ is not a leaf} \\
\frac{d}{dt}x_{\boldsymbol{l}}(t) &= F_{\boldsymbol{l}}f_{\boldsymbol{l}}(t) - F_{\boldsymbol{l}}\mu_{\boldsymbol{l}}(t), \text{ if } \boldsymbol{l} \text{ is a leaf} \qquad (7) \\
\sum_{\boldsymbol{l}\in\mathcal{L}} \frac{d}{dt}x_{\boldsymbol{l}}(t) &= \sum_{\boldsymbol{l}\in\mathcal{L}} F_{\boldsymbol{l}}f_{\boldsymbol{l}}(t) - \sum_{l=1}^{C} F_l\mu_l(t) \qquad (8)
\end{aligned}$$

We now briefly show how we can derive (7) for the case when $\boldsymbol{l}$ is a leaf. From (1), we have

$$\frac{1}{B}X_{\boldsymbol{l}}(B(T+t)) = \frac{1}{B}\sum_{\tau=0}^{B(T+t)} (A_{\boldsymbol{l}}(\tau) - E_{\boldsymbol{l}}(\tau)) + O(\frac{1}{B}).$$

The term $O(\frac{1}{B})$ accounts for the terms $\frac{1}{B}\left(X_{\boldsymbol{l}}(0) - A_{\boldsymbol{l}}(B(T+t)) + E_{\boldsymbol{l}}(B(T+t))\right)$. Taking the limit as $B \to \infty$ along the subsequence that gives us the FSP, we have $x_{\boldsymbol{l}}(t) = a_{\boldsymbol{l}}(t) - e_{\boldsymbol{l}}(t)$. Differentiating, we obtain (7).

The following proposition captures fundamental constraints in a converge-cast for the FSPs under any algorithm.

Proposition 1: For any scheduling algorithm, any FSP $(\boldsymbol{a}(t), \boldsymbol{x}(t), \boldsymbol{e}(t))$ must satisfy the following constraints for all regular time t.

Interference constraint equations:

$$\begin{aligned}
\sum_{l=1}^{C} \mu_l(t) &\leq 1 \qquad (9) \\
\sum_{l=1}^{C(\boldsymbol{l})} \mu_{<\boldsymbol{l},l>}(t) + \mu_{\boldsymbol{l}}(t) &\leq 1 \text{ for all interior nodes } \boldsymbol{l} \qquad (10) \\
\mu_{\boldsymbol{l}}(t) &\in [0,1] \text{ for all nodes } \boldsymbol{l} \qquad (11)
\end{aligned}$$

Flow constraint equations:

$$\begin{aligned}
\mu_{\boldsymbol{l}}(t) &\leq f_{\boldsymbol{l}}(t) + \sum_{l=1}^{C(\boldsymbol{l})} \frac{F_{<\boldsymbol{l},l>}}{F_{\boldsymbol{l}}}\mu_{<\boldsymbol{l},l>}(t) \qquad (12) \\
&\quad \text{if } x_{\boldsymbol{l}}(t) = 0 \ \& \ \boldsymbol{l} \text{ is an interior node} \\
\mu_{\boldsymbol{l}}(t) &\leq f_{\boldsymbol{l}}(t) \text{ if } x_{\boldsymbol{l}}(t) = 0 \ \& \ \boldsymbol{l} \text{ is a leaf} \qquad (13)
\end{aligned}$$

Remark: If we think of $\mu_{\boldsymbol{l}}(t)$ as the fraction of time that link $\boldsymbol{l}$ is activated, Equations (9)-(11) state that for any set of interfering links, the sum of the fractions of time that each link in the set is activated must be less than 1. Equations (12) and (13) state that when the queue backlog $x_{\boldsymbol{l}}(t)$ at node $\boldsymbol{l}$ is 0, the net flow of data into the node must exceed the flow out of the node.

Proof: First we prove the interference constraint equations (9)-(11). Consider the root node. Due to interference, only one of the links $1, \ldots, C$ can be active in a time-slot. Hence,

$$\sum_{l=1}^{C} \frac{E_l(\tau)}{F_l} \leq 1.$$

for any time-slot τ. Summing both sides over $\tau = B(T+t)$ to $B(T+t+\delta)$ and dividing both sides by B, we can derive

$$\sum_{l=1}^{C} \frac{e_l^B(t+\delta) - e_l^B(t)}{F_l} \leq \delta \tag{14}$$

where we have used the fluid scaling notation (6). Note that this inequality holds for all B and T, and for all $e_l^B(\cdot)$.

By definition of FSP, there exists (for some $T > 0$) a sequence $(\boldsymbol{a}^B(t), \boldsymbol{x}^B(t), \boldsymbol{e}^B(t))$ that converges to $(\boldsymbol{a}(t), \boldsymbol{x}(t), \boldsymbol{e}(t))$. Since $(\boldsymbol{a}^B(t), \boldsymbol{x}^B(t), \boldsymbol{e}^B(t))$ satisfies (14), we must have

$$\sum_{l=1}^{C} \frac{e_l(t+\delta) - e_l(t)}{F_l} \leq \delta$$

Dividing both sides by δ and taking the limit as $\delta \to 0$, we obtain

$$\sum_{l=1}^{C} \mu_l(t) \quad \leq \quad 1.$$

Next, consider any interior node $\boldsymbol{l}$. Due to interference, only one of the links $\boldsymbol{l}, <\boldsymbol{l}, 1>, \ldots, <\boldsymbol{l}, C(\boldsymbol{l})>$ can be active in a time-slot τ. We have

$$\sum_{l=1}^{C(\boldsymbol{l})} \frac{E_{<\boldsymbol{l},l>}(\tau)}{F_{<\boldsymbol{l},l>}} + \frac{E_{\boldsymbol{l}}(\tau)}{F_{\boldsymbol{l}}} \leq 1.$$

Applying similar algebraic operations as before, we can derive

$$\sum_{l=1}^{C(\boldsymbol{l})} \mu_{<\boldsymbol{l},l>}(t) + \mu_{\boldsymbol{l}}(t) \quad \leq \quad 1$$

Similarly, for any node $\boldsymbol{l}$ including the root, we have

$$0 \leq \frac{E_{\boldsymbol{l}}(\tau)}{F_{\boldsymbol{l}}} \leq 1$$

for any time-slot τ. From this, we can show that

$$0 \leq \mu_{\boldsymbol{l}}(t) \leq 1$$

The flow constraints follow from (7) and the fact that if $x_{\boldsymbol{l}}(t) = 0$ then $\frac{d}{dt} x_{\boldsymbol{l}}(t) \geq 0$. ■

Define the Lyapunov function $V(\boldsymbol{x}(t)) \triangleq \sum_{\boldsymbol{l} \in \mathcal{L}} x_{\boldsymbol{l}}(t)$. We will use the results of [13] to prove that the P-TREE algorithm is optimal in terms of maximizing the large deviations decay rate. Specifically, define I_π and J_π (correspondingly, I_{p-t} and J_{p-t}) to be the quantities (4) and (5) when the system is operating under scheduling algorithm π (correspondingly, under P-TREE). We will show that $J_{p-t} \geq I_\pi$ for all algorithms π. Hence, the fastest rate of decay of $\mathbf{P}[\sum_{\boldsymbol{l} \in \mathcal{L}} X_{\boldsymbol{l}}(0) \geq B]$ is that obtained under the P-TREE algorithm. Recall for future reference that $\mathbf{P}[\sum_{\boldsymbol{l} \in \mathcal{L}} X_{\boldsymbol{l}}(0) \geq B] = \mathbf{P}[V(\boldsymbol{x}^B(0)) \geq 1]$. We state this formally as follows.

Proposition 2: The P-TREE algorithm attains the optimal decay rate, i.e., for any scheduling algorithm π, we have

$$\limsup_{B \to \infty} \frac{1}{B} \log \left(\mathbf{P}^{p-t} \left[\sum_{l \in \mathcal{L}} X_l(0) \geq B \right] \right)$$
$$\leq \liminf_{B \to \infty} \frac{1}{B} \log \left(\mathbf{P}^{\pi} \left[\sum_{l \in \mathcal{L}} X_l(0) \geq B \right] \right),$$

i.e., $J_{p-t} \geq I_{\pi}$.

To prove Proposition 2, we use the result of Theorem 8 from [13] which we restate here for reference.

Theorem 3: Let π_0 be a scheduling policy that satisfies Assumptions 1, 2, 3, 4, 5 and 6 (see appendix A). Let π be any scheduling policy, then $\limsup_{B\to\infty} \frac{1}{B}\log(\mathbf{P}^{\pi_0}[V(\boldsymbol{x}^B(0)) \geq 1]) \leq \liminf_{B\to\infty} \frac{1}{B}\log(\mathbf{P}^{\pi}[V(\boldsymbol{x}^B(0)) \geq 1])$.

Assumptions 1, 2, 3, 5 and 6 are stated in appendix A along with the proof of Lemma 4 which verifies that the P-TREE algorithm in fact satisfies the stated assumptions.

Lemma 4: The P-TREE algorithm and Lyapunov function $V(\cdot)$ satisfy Assumptions 1, 2, 3, 5 and 6 mentioned in [13].

Assumption 4 is stated below.

Assumption 4: For any FSP $(\boldsymbol{a}(t), \boldsymbol{x}(t), \boldsymbol{e}(t))$, the algorithm π_0 satisfies the following for all regular time t:

$$\begin{aligned} \frac{d}{dt}V(\boldsymbol{x}(t)) = \min_{\hat{\boldsymbol{\mu}}} \quad & \frac{\partial}{\partial\tau}V(\boldsymbol{x}(t) + (\boldsymbol{f}(t) - \hat{\boldsymbol{\mu}})\tau)\Big|_{\tau=0} \\ \text{subject to} \quad & \boldsymbol{x}(t), \boldsymbol{f}(t), \hat{\boldsymbol{\mu}} \text{ satisfy FSP constraints} \\ & \text{in Proposition 1} \end{aligned}$$

Assumption 4 states that the scheduling algorithm π_0 minimizes the drift, $\frac{d}{dt}V(\boldsymbol{x}(t))$, at each point in time over all possible scheduling algorithms. This assumption is the key assumption that the P-TREE algorithm needs to satisfy for the result Proposition 2 to hold. The rest of this section will be dedicated to verifying that the P-TREE algorithm in fact satisfies this assumption.

First we define an optimization problem to obtain a lower bound on the drift of $V(\boldsymbol{x}(t))$, $\frac{d}{dt}V(\boldsymbol{x}(t))$, under any scheduling policy. Note that the drift is given by $\frac{d}{dt}V(\boldsymbol{x}(t)) = \sum_{\boldsymbol{l}\in\mathcal{L}} \frac{d}{dt}x_{\boldsymbol{l}}(t)$ where $\sum_{\boldsymbol{l}\in\mathcal{L}} \frac{d}{dt}x_{\boldsymbol{l}}(t)$ is given in (8).

It is easy to see that the following optimization problem bounds from below the drift, $\frac{d}{dt}V(\boldsymbol{x}(t))$, of any FSP$(\boldsymbol{a}(t), \boldsymbol{x}(t), \boldsymbol{e}(t))$ of any scheduling algorithm for regular time t.

$$\text{optA}(\boldsymbol{f}(t), \boldsymbol{x}(t)): \tag{15}$$

$$\begin{aligned} \min_{\boldsymbol{\mu}(t)} \quad & \sum_{\boldsymbol{l}\in\mathcal{L}} F_{\boldsymbol{l}} f_{\boldsymbol{l}}(t) - \sum_{l=1}^{C} F_l \mu_l(t) \\ \text{sub to} \quad & \sum_{l=1}^{C} \mu_l(t) \leq 1 \\ & \sum_{l=1}^{C(\boldsymbol{l})} \mu_{<\boldsymbol{l},l>}(t) + \mu_{\boldsymbol{l}}(t) \leq 1 \text{ for all interior} \\ & \qquad \text{nodes } \boldsymbol{l}. \\ & \mu_{\boldsymbol{l}}(t) \in [0,1] \text{ for all nodes } \boldsymbol{l}. \\ & \mu_{\boldsymbol{l}}(t) \leq f_{\boldsymbol{l}}(t) + \sum_{l=1}^{C(\boldsymbol{l})} \frac{F_{<\boldsymbol{l},l>}}{F_{\boldsymbol{l}}} \mu_{<\boldsymbol{l},l>}(t) \text{ if} \\ & \qquad x_{\boldsymbol{l}}(t) = 0 \ \& \ \boldsymbol{l} \text{ is an interior node.} \\ & \mu_{\boldsymbol{l}}(t) \leq f_{\boldsymbol{l}}(t) \text{ if } x_{\boldsymbol{l}}(t) = 0 \ \& \ \boldsymbol{l} \text{ is a leaf.} \end{aligned}$$

To see that (15) provides a lower bound for the drift, note that the objective function of the optimization problem is $\frac{d}{dt}V(\boldsymbol{x}(t))$ and the constraints are the set of inequalities satisfied by any FSP as stated in Proposition 1. By minimizing over all possible values of $\boldsymbol{\mu}(t)$, we obtain a lower bound on the drift under any algorithm.

The following Lemma formally states that the P-TREE algorithm satisfies Assumption 4.

Lemma 5: For any FSP $(\boldsymbol{a}(t), \boldsymbol{x}(t), \boldsymbol{e}(t))$ of the P-TREE algorithm, the drift is given by $\frac{d}{dt}V(\boldsymbol{x}(t)) = \text{optA}(\boldsymbol{f}(t), \boldsymbol{x}(t))$.

Before we present the proofs, we briefly illustrate where lies the difficulty in verifying that the P-TREE algorithm satisfies Assumption 4. Note that the drift of the Lyapunov function $\sum_{\boldsymbol{l}\in\mathcal{L}} x_{\boldsymbol{l}}$ is equal to $\sum_{\boldsymbol{l}\in\mathcal{L}} F_{\boldsymbol{l}} f_{\boldsymbol{l}} - F_{\boldsymbol{l}}\mu_{\boldsymbol{l}}$. Suppose that link 1 is at depth 1, and it has the largest rate among all links at depth 1. At a given point t in an FSP, suppose that $x_1(t) > 0$, which means that in the original discrete time system the backlog of link 1 is very large. Then it is easy to see that P-TREE minimizes the drift because it will always activate link 1, and hence $\mu_1(t) = 1$. What is more complicated, however, is when $x_1(t) = 0$. In this case, the backlog of link 1 in the original discrete-time system is close to (but not always equal to) zero. Hence, under the P-TREE algorithm, link l will be served for some fraction of time. The exact fraction will depend on the services at its children links that feed packets into link 1. The situation will become even more complicated when these children links $\boldsymbol{l}$ in turn have $x_{\boldsymbol{l}}(t) = 0$ in the FSP. Hence, in order to prove that P-TREE algorithm minimizes the drift at time t in FSP, we must carefully account for all possible combinations of the values $x_{\boldsymbol{l}}(t)$ (being zero or strictly positive), which makes the analysis quite complicated. However, we will develop an important result (Proposition 7) that reveals an interesting structure of the dynamics of the P-TREE algorithm, which successfully addresses the above difficulty.

The rest of the section is dedicated to proving Lemma 5 and is divided into two subsections. In subsection V-A, we derive properties of the FSP under the P-TREE algorithm. Then, in subsection V-B, we show that the drift under the P-TREE algorithm achieves the value of optA$(\boldsymbol{f}(t), \boldsymbol{x}(t))$. Since optA$(\boldsymbol{f}(t), \boldsymbol{x}(t))$ is a lower bound on the drift of any scheduling algorithm, this implies that the P-TREE algorithm satisfies Assumption 4 and that Lemma 5 holds.

A. Properties of FSPs under P-Tree algorithm

The following lemma proves that whenever the backlog for a link is positive or if the backlog is zero but is growing at a positive rate, then under the P-TREE algorithm this link must receive all remaining service possible after service has been assigned to the higher priority links.

Lemma 6: Any FSP $(\boldsymbol{a}(t), \boldsymbol{x}(t), \boldsymbol{e}(t))$ under the P-TREE algorithm satisfies the following for regular time t:
For $l = 1, \ldots, C$, if $x_l(t) > 0$ or if $x_l(t) = 0$ and $\frac{d}{dt}x_l(t) > 0$, then $\mu_l(t) = 1 - \sum_{i=1}^{l-1}\mu_i(t)$.

For any interior node $\boldsymbol{l}$, and $l = 1, \ldots, C(\boldsymbol{l})$ if $x_{<\boldsymbol{l},l>}(t) > 0$ or $x_{<\boldsymbol{l},l>}(t) = 0$ and $\frac{d}{dt}x_{<\boldsymbol{l},l>}(t) > 0$, then $\mu_{<\boldsymbol{l},l>}(t) = 1 - \mu_{\boldsymbol{l}}(t) - \sum_{i=1}^{l-1}\mu_{<\boldsymbol{l},i>}(t)$

Proof: We will prove the equation for $l = 1, \ldots, C$. The proof for $l = 1, \ldots, C(\boldsymbol{l})$, where $\boldsymbol{l}$ is an interior node, follows similar arguments and hence is omitted.

Consider $x_l(t) > 0$. Since $\boldsymbol{x}(t)$ is Lipschitz continuous and the convergence of $(\boldsymbol{a}^B(t), \boldsymbol{x}^B(t), \boldsymbol{e}^B(t))$ to FSP $(\boldsymbol{a}(t), \boldsymbol{x}(t), \boldsymbol{e}(t))$ is u.o.c (uniform over compact intervals), there exists $\epsilon > 0$, $B_0 > 0$ and $\delta_0 > 0$ such that for $B > \max\{B_0, F_l/\epsilon\}$ and $0 < \delta < \delta_0$, $X_l(B(t+\delta)) > B\epsilon > F_l$. Hence, for time-slot τ in the interval $[Bt, B(t+\delta_0)]$, the P-TREE algorithm will activate link l whenever none of the higher priority links $1, \ldots, l-1$ is activated. Hence, we have

$$\frac{E_l(\tau)}{F_l} = 1 - \sum_{i=1}^{l-1}\frac{E_i(\tau)}{F_i}.$$

Summing over the time interval $[Bt, B(t+\delta)]$ and dividing both sides by B, we have

$$\frac{e_l^B(t+\delta) - e_l^B(t)}{F_l} = \delta - \sum_{i=1}^{l-1}\frac{e_i^B(t+\delta) - e_i^B(t)}{F_i}.$$

Taking the limit as $B \to \infty$, dividing both sides by δ and further taking the limit as $\delta \to 0$, we obtain $\mu_l(t) = 1 - \sum_{i=1}^{l-1} \mu_i(t)$

Now consider the case when $x_{\boldsymbol{l}}(t) = 0$ and $\frac{d}{dt}x_l(t) > 0$. This means that there exists $\epsilon > 0$ and $\delta_0 > 0$ such that for $0 < \delta < \delta_0$, $x_l(t+\delta) > \epsilon\delta$. The fact that $x_l^B(t) \to x_l(t)$ u.o.c implies that for any $\tilde{\epsilon} \in [0, \epsilon\delta_0/4]$ we can find B_0 such that for $B > \max\{B_0, F_l/\tilde{\epsilon}\}$, $x_l^B(t+\delta) > (\epsilon\delta - \tilde{\epsilon})^+$ or in other words, $X_l(B(t+\delta)) > B(\epsilon\delta - \tilde{\epsilon})^+$. Note that for $\delta \in [2\tilde{\epsilon}/\epsilon, \delta_0]$, we thus have $X_l(B(t+\delta)) > F_l$ and hence the priority algorithm will activate link l whenever none of the higher priority links $1, \ldots, l-1$ is activated. Hence, during timeslots $\tau \in [B(t + 2\tilde{\epsilon}/\epsilon), B(t+\delta_0)]$ we have

$$\frac{E_l(\tau)}{F_l} = 1 - \sum_{i=1}^{l-1} \frac{E_i(\tau)}{F_i}.$$

Take any $\delta \in [2\tilde{\epsilon}/\epsilon, \delta_0]$. Consider

$$\begin{aligned}
& \frac{e_l^B(t+\delta) - e_l^B(t)}{F_l} \\
= \; & \frac{1}{B} \sum_{\tau=Bt}^{B(t+\delta)} \frac{E_l(\tau)}{F_l} \\
\geq \; & \frac{1}{B} \sum_{\tau=B(t+2\tilde{\epsilon}/\epsilon)}^{B(t+\delta)} \left(1 - \sum_{i=1}^{l-1} \frac{E_i(\tau)}{F_i}\right) \\
\geq \; & \delta - 2\tilde{\epsilon}/\epsilon - \sum_{i=1}^{l-1} \frac{e_i^B(t+\delta) - e_i^B(t+2\tilde{\epsilon}/\epsilon)}{F_i}.
\end{aligned}$$

Taking the limit as $B \to \infty$, we get

$$\frac{e_l(t+\delta) - e_l(t)}{F_l} \geq \delta - 2\tilde{\epsilon}/\epsilon - \sum_{i=1}^{l-1} \frac{e_i(t+\delta) - e_i(t+2\tilde{\epsilon}/\epsilon)}{F_i}.$$

Since the above is true for any $\tilde{\epsilon} \in [0, \epsilon\delta_0/4]$, we can take the limit as $\tilde{\epsilon} \to 0$. Hence, for any $\delta \in (0, \delta_0]$, we have

$$\frac{e_l(t+\delta) - e_l(t)}{F_l} \geq \delta - \sum_{i=1}^{l-1} \frac{e_i(t+\delta) - e_i(t)}{F_i}.$$

Dividing by δ and taking the limit as $\delta \to 0$, we obtain $\mu_l(t) \geq 1 - \sum_{i=1}^{l-1} \mu_i(t)$. Recall from Proposition 1 that $\mu_l(t) \leq 1 - \sum_{i=1}^{l-1} \mu_i(t)$. This concludes the proof. ■

We can now prove the following proposition which states the following. If the backlog at a link is positive, then the link receives all remaining service possible after service has been allocated to higher priority links. If the backlog is zero, then it receives the smaller of two quantities. One is the amount of data flowing in from the children and the other is the maximum amount of service the link can receive after taking into account the amount of service given to higher priority links.

Proposition 7: Any FSP $(\boldsymbol{a}(t), \boldsymbol{x}(t), \boldsymbol{e}(t))$ of P-TREE algorithm satisfies the following for all regular time t

For $l = 1, \ldots, C$:

If $x_l(t) > 0$, then the following holds

$$\mu_l(t) = 1 - \sum_{i=1}^{l-1} \mu_i(t)$$

If $x_l(t) = 0$, then the following holds

$$\mu_l(t) = \begin{cases} \min\Big(1 - \sum_{i=1}^{l-1} \mu_i(t)\ , \\ \qquad f_l(t) + \sum_{j=1}^{C(l)} \mu_{<l,j>}(t) \frac{F_{<l,j>}}{F_l}\Big) \\ \qquad \text{if } l \text{ is not a leaf.} \\ \min\Big(1 - \sum_{i=1}^{l-1} \mu_i(t), f_l(t)\Big) \\ \qquad \text{if } l \text{ is a leaf.} \end{cases}$$

For any interior node $\boldsymbol{l}$, $l = 1, \ldots, C(\boldsymbol{l})$:

If $x_{<\boldsymbol{l},l>}(t) > 0$, then the following holds

$$\mu_{<\boldsymbol{l},l>}(t) = 1 - \mu_{\boldsymbol{l}}(t) - \sum_{i=1}^{l-1} \mu_{<\boldsymbol{l},j>}(t)$$

If $x_{<\boldsymbol{l},l>}(t) = 0$, then the following holds

$$\mu_{<\boldsymbol{l},l>}(t) = \begin{cases} \min\Big(1 - \mu_{\boldsymbol{l}}(t) - \sum_{i=1}^{l-1} \mu_{<\boldsymbol{l},j>}(t), \\ \quad f_{<\boldsymbol{l},l>}(t) + \sum_{j=1}^{C(<\boldsymbol{l},l>)} \mu_{<\boldsymbol{l},l,j>}(t) \frac{F_{<\boldsymbol{l},l,j>}}{F_{<\boldsymbol{l},l>}})\Big) \\ \qquad \text{if } <\boldsymbol{l}, l> \text{ is not a leaf.} \\ \min\Big(1 - \mu_{\boldsymbol{l}}(t) - \sum_{i=1}^{l-1} \mu_{<\boldsymbol{l},j>}(t), f_{<\boldsymbol{l},l>}(t)\Big) \\ \qquad \text{if } <\boldsymbol{l}, l> \text{ is a leaf.} \end{cases}$$

Remark: The idea expressed by the proposition is the following. Consider link l connected to the root (the first set of equations). If $x_l(t) > 0$, then under any algorithm, the link l can at most be assigned all the remaining service after service has been assigned to higher priority links. Hence, we will have the inequality $\mu_l(t) \leq 1 - \sum_{i=1}^{l-1} \mu_i(t)$. What the above proposition says is that under the P-TREE algorithm, we will have strict equality. Loosely speaking, this means that the P-TREE algorithm uses up all the service. On the other hand, if $x_l(t) = 0$, then the amount of service given to link l will be constrained by the additional requirement that the out-flow at a node can not exceed the in-flow into the node (see Proposition 1). For example, if link l is also a leaf node, then under any algorithm, the service given to link l will be determined by which of the two $1 - \sum_{i=1}^{l-1} \mu_i(t)$ and $f_l(t)$ is smaller. Hence we will have the inequality $\mu_l(t) \leq \min\Big(1 - \sum_{i=1}^{l-1} \mu_i(t), f_l(t)\Big)$. Again, what the proposition says is that the P-TREE algorithm attains strict equality. A similar intuition applies to other parts of the proposition.

Proof: First, consider the links connecting to the root, $l = 1, \ldots, C$. Consider $x_l(t) > 0$. Using Lemma 6, we obtain $\mu_l(t) = 1 - \sum_{i=1}^{l-1} \mu_i(t)$.

Now, consider the case $x_l(t) = 0$ and assume node l is not a leaf. Lets assume that

$$\begin{aligned} \mu_l(t) \quad \neq \quad & \min\Bigg(1 - \sum_{i=1}^{l-1} \mu_i(t)\ , \\ & \lambda_l(t) + \sum_{j=1}^{C(l)} \mu_{<l,j>}(t) \frac{F_{<l,j>}}{F_l}\Bigg). \end{aligned} \tag{16}$$

Due to Proposition 1, we must have $\mu_l(t) \leq \min\left(1-\sum_{i=1}^{l-1}\mu_i(t), \lambda_l(t)+\sum_{j=1}^{C(l)}\mu_{<l,j>}(t)\frac{F_{<l,j>}}{F_l}\right)$. Hence the only possibility is $\mu_l(t)$ is less than the right-hand-side of (16). There must then exist $\gamma > 0$ such that $\mu_l(t) < \min\left(1-\sum_{i=1}^{l-1}\mu_i(t), \lambda_l(t)+\sum_{j=1}^{C(l)}\mu_{<l,j>}(t)\right.$ γ, or in other words,

$$\begin{aligned} & \min\left(1-\sum_{i=1}^{l-1}\mu_i(t), \lambda_l(t)+\sum_{j=1}^{C(l)}\mu_{<l,j>}(t)\frac{F_{<l,j>}}{F_l}\right) \\ \geq \quad & \mu_l(t)+\gamma \end{aligned}$$

We will now show that this leads to $\frac{d}{dt}x_l(t) > 0$. Recall from (7) that $\frac{d}{dt}x_l(t) = F_l f_l(t) + \sum_{j=1}^{C(l)} F_{<l,j>}\mu_{<l,j>}(t) - F_l\mu_l(t)$. We have

$$\begin{aligned} \frac{d}{dt}\frac{x_l(t)}{F_l} \quad \geq \quad & \min\left(1-\sum_{i=1}^{l-1}\mu_i(t), f_l+\sum_{j=1}^{C(l)}\mu_{<l,j>}\frac{F_{<l,j>}}{F_l}\right) \\ & -\mu_l(t) \\ \geq \quad & \gamma > 0 \end{aligned}$$

Hence, we have $\frac{d}{dt}x_l(t) > 0$. The consequence of this are the following: by (7) we have, $\mu_l(t) < f_l(t)+\sum_{j=1}^{C(l)}\mu_{<l,j>}(t)\frac{F_{<l,j>}}{F_l}$ and by Lemma 6 we have, $\mu_l(t) = 1-\sum_{i=1}^{l-1}\mu_i(t)$. We then have a contradiction with our initial assumption (16) and hence it must be true that $\mu_l(t) = \min\left(1-\sum_{i=1}^{l-1}\mu_i(t), f_l(t)+\sum_{j=1}^{C(l)}\mu_{<l,j>}(t)\frac{F_{<l,j>}}{F_l}\right)$.

The rest of the cases in the statement of the proposition can be proved using similar ideas as outlined above. We omit details for brevity. ■

B. Proof of Lemma 5

So far, we have only shown that the FSP of the P-TREE algorithm satisfies certain properties, which correspond to different combinations of the value $x_{\boldsymbol{l}}(t)$ (being zero or strictly positive). To prove Prop. 2, we need to verify that P-TREE minimizes the drift at each time in every FSPs. More precisely, we need to prove that any FSP $(\boldsymbol{a}(t), \boldsymbol{x}(t), \boldsymbol{e}(t))$ of the P-TREE algorithm has drift equal to optA($\boldsymbol{f}(t), \boldsymbol{x}(t)$), i.e. $\boldsymbol{\mu}(t)$ is an optimizer for optA($\boldsymbol{f}(t), \boldsymbol{x}(t)$) at every t.

Our strategy is to prove by contradiction. Assume that $\boldsymbol{\mu}(t)$ does not optimize optA($\boldsymbol{f}(t), \boldsymbol{x}(t)$). Then, there must exist a change in service $\boldsymbol{\delta}$ such that $\boldsymbol{\mu}(t)+\boldsymbol{\delta}$ provides a better value for the objective function of optA($\boldsymbol{f}(t), \boldsymbol{x}(t)$) while at the same time satisfying the constraints of optA($\boldsymbol{f}(t), \boldsymbol{x}(t)$). Note that the difference between the values of the objective function for $\boldsymbol{\mu}(t)$ and for $\boldsymbol{\mu}(t)+\boldsymbol{\delta}$ is equal to $\sum_{l=1}^{C} F_l\delta_l$. Hence, it suffices to show that no $\boldsymbol{\delta}$ can produce $\sum_{l=1}^{C} F_l\delta_l > 0$ while still satisfying the constraints of optA($\boldsymbol{f}(t), \boldsymbol{x}(t)$). This is proved in Proposition 8.

Proposition 8: For all $\boldsymbol{\delta}$ such that $\boldsymbol{\mu}(t)+\boldsymbol{\delta}$ satisfies the constraint equations for optA($\boldsymbol{f}(t), \boldsymbol{x}(t)$), we must have $\sum_{l=1}^{C} F_l\delta_l \leq 0$

Before we can prove Proposition 8, we will need to derive certain properties of $\boldsymbol{\delta}$ based on the assumption that $\boldsymbol{\mu}(t)+\boldsymbol{\delta}$ satisfies the constraints of optA.

Lemma 9: If $\boldsymbol{\mu}(t)+\boldsymbol{\delta}$ satisfies the constraints of optA($\boldsymbol{f}(t), \boldsymbol{x}(t)$), then $\boldsymbol{\delta}$ satisfies the following:

For $l = 1, \ldots, C$:

$$\delta_l \leq \begin{cases} \max\left(-\sum_{i=1}^{l-1}\delta_i, \sum_{j=1}^{C(l)}\delta_{<l,j>}\frac{F_{<l,j>}}{F_l}\right) \\ \qquad \text{if } l \text{ is not a leaf.} \\ \max\left(-\sum_{i=1}^{l-1}\delta_i, 0\right) \\ \qquad \text{if } l \text{ is a leaf.} \end{cases}$$

For any interior node $\boldsymbol{l}$ and $l = 1, \ldots, C(\boldsymbol{l})$

$$\delta_{<\boldsymbol{l},l>} \leq \begin{cases} \max\Big(-\delta_{\boldsymbol{l}} - \sum_{i=1}^{l-1}\delta_{<\boldsymbol{l},j>}, \\ \qquad \sum_{j=1}^{C(<\boldsymbol{l},l>)}\delta_{<\boldsymbol{l},l,j>}\frac{F_{<\boldsymbol{l},l,j>}}{F_{<\boldsymbol{l},l>}}\Big) \\ \qquad\qquad \text{if } <\boldsymbol{l},l> \text{ is not a leaf.} \\ \max\left(-\delta_{\boldsymbol{l}} - \sum_{i=1}^{l-1}\delta_{<\boldsymbol{l},j>}, 0\right) \\ \qquad\qquad \text{if } <\boldsymbol{l},l> \text{ is a leaf.} \end{cases}$$

Remark: Note that this is a critical property for the overall proof because it holds regardless of the value of $x_{\boldsymbol{l}}(t)$, which as we discussed before, has been the main source of complexity. This Lemma expresses the following intuition: Recall from Proposition 7 that the P-TREE algorithm uses up all the service available. In such a situation, the increase in service δ_l for any link l is constrained by two factors. We must either sacrifice service (i.e. reduce δ_j) at higher priority links $j = 1, \ldots, l-1$ or increase service to the children of l. Hence, the change in service δ_l can at most be $\max\left(-\sum_{i=1}^{l-1}\delta_i, \sum_{j=1}^{C(l)}\delta_{<l,j>}\frac{F_{<l,j>}}{F_l}\right)$. Note that if a link is a leaf, then it does not have children and hence the second factor does not appear.

Proof: Consider link $l = 1, \ldots, C$. Since $\mu_l(t) + \delta_l$ is feasible, from the constraint equations of optA, we can derive

$$\mu_l(t) + \delta_l \leq 1 - \sum_{i=1}^{l-1}(\mu_i(t) + \delta_i) \tag{17}$$

$$\mu_l(t) + \delta_l \leq f_l(t) + \sum_{j=1}^{C(l)}(\mu_{<l,j>} + \delta_{<l,j>})\frac{F_{<l,j>}}{F_l} \tag{18}$$

If $x_l(t) > 0$, then by Proposition 7, we have $\mu_l(t) = 1 - \sum_{i=1}^{l-1}\mu_i(t)$. This with (17) proves $\delta_l \leq -\sum_{i=1}^{l-1}\delta_i$ and hence $\delta_l \leq \max\left(-\sum_{i=1}^{l-1}\delta_i, \sum_{j=1}^{C(l)}\delta_{<l,j>}\frac{F_{<l,j>}}{F_l}\right)$.

If $x_l(t) = 0$ and l is not a leaf, then by Proposition 7, we have $\mu_l(t) = \min\left(1 - \sum_{i=1}^{l-1}\mu_i(t), f_l(t) + \sum_{j=1}^{C(l)}\mu_{<l,j>}\frac{F_{<l,j>}}{F_l}\right)$. If $1 - \sum_{i=1}^{l-1}\mu_i(t) \leq f_l(t) + \sum_{j=1}^{C(l)}\mu_{<l,j>}\frac{F_{<l,j>}}{F_l}$, we have $\mu_l(t) = 1 - \sum_{i=1}^{l-1}\mu_i(t)$. This with (17) implies $\delta_l \leq -\sum_{i=1}^{l-1}\delta_i$. On the other hand, if $1 - \sum_{i=1}^{l-1}\mu_i(t) > f_l(t) + \sum_{j=1}^{C(l)}\mu_{<l,j>}\frac{F_{<l,j>}}{F_l}$, we have $\mu_l(t) = f_l(t) + \sum_{j=1}^{C(l)}\mu_{<l,j>}\frac{F_{<l,j>}}{F_l}$. This with (18) implies $\delta_l \leq \sum_{j=1}^{C(l)}\delta_{<l,j>}\frac{F_{<l,j>}}{F_l}$. Hence we conclude $\delta_l \leq \max\left(-\sum_{i=1}^{l-1}\delta_i, \sum_{j=1}^{C(l)}\delta_{<l,j>}\frac{F_{<l,j>}}{F_l}\right)$.

The other cases can be proved using the ideas outlined above. ■

Let Ω be a number that upper bounds $\frac{F_{<\boldsymbol{l},l>}}{F_{\boldsymbol{l}}}$ for all links $\boldsymbol{l}$ and $l = 1, \ldots, C(\boldsymbol{l})$.

Lemma 10: If $\boldsymbol{\mu}(t) + \boldsymbol{\delta}$ satisfies the constraints for optA($\boldsymbol{f}(t), \boldsymbol{x}(t)$), then $\boldsymbol{\delta}$ satisfies the following:
Consider link $\boldsymbol{l}$ such that for $l = 1, \ldots, C(\boldsymbol{l})$, $\delta_{<\boldsymbol{l},l>} \leq \max(-\delta_{\boldsymbol{l}} - \sum_{j=1}^{l-1}\delta_{<\boldsymbol{l},j>}, 0)$. Then $\sum_{l=1}^{C(\boldsymbol{l})}\delta_{<\boldsymbol{l},l>}\frac{F_{<\boldsymbol{l},l>}}{F_{\boldsymbol{l}}} \leq \max(-\delta_{\boldsymbol{l}}\Omega, 0)$.
Remark: The significance of this Lemma is the following: The assumption $\delta_{<\boldsymbol{l},l>} \leq \max(-\delta_{\boldsymbol{l}} - \sum_{j=1}^{l-1}\delta_{<\boldsymbol{l},j>}, 0)$ captures the requirement that the increase in service to a child link $<\boldsymbol{l},l>$ can only come at a loss in service to higher priority links

$\boldsymbol{l}, <\boldsymbol{l}, 1>, \ldots, <\boldsymbol{l}, l-1>$ (for example, the requirement holds when all the child nodes of $\boldsymbol{l}$ were leaf nodes). This lemma states that if all the children links $<\boldsymbol{l}, l>$ are subject to the above requirement, then a positive increase in the service of the child nodes $\sum_{l=1}^{C(\boldsymbol{l})} \delta_{<\boldsymbol{l},l>} \frac{F_{<\boldsymbol{l},l>}}{F_{\boldsymbol{l}}}$ can only come from a reduction in service, $-\delta_{\boldsymbol{l}}$, to the parent node $\boldsymbol{l}$. The consequence of this Lemma will be that the parent link $\boldsymbol{l}$ will also be subject to the same requirement that an increase in service to $\boldsymbol{l}$ can only come at a loss in service to links that have higher priority than $\boldsymbol{l}$ (see Lemma 11).

Proof: To prove the lemma, it is enough to show that for any mathematical quantities $\delta_{\boldsymbol{l}}$ and $\delta_{<\boldsymbol{l},1>}, \ldots, \delta_{<\boldsymbol{l},C(\boldsymbol{l})>}$ which satisfy the inequalities

$$\delta_{<\boldsymbol{l},l>} \leq \max(-\delta_{\boldsymbol{l}} - \sum_{j=1}^{l-1} \delta_{<\boldsymbol{l},j>}, 0),$$

for $l = 1, \ldots, C(\boldsymbol{l})$, and any non-increasing, non-negative sequence $\{F_{<\boldsymbol{l},l>}\}$ bounded by $\Omega F_{\boldsymbol{l}}$, the following is true

$$\sum_{l=1}^{C(\boldsymbol{l})} \delta_{<\boldsymbol{l},l>} \frac{F_{<\boldsymbol{l},l>}}{F_{\boldsymbol{l}}} \leq \max(-\delta_{\boldsymbol{l}}\Omega, 0).$$

We emphasize that this is a purely mathematical result and that allowing $\{F_{<\boldsymbol{l},l>}\}$ to represent various sequences is simply a trick to shorten the proof. It does not mean that we consider various systems with different values for the link capacities.

We will prove this by induction. By our assumption, we know that $\delta_{<\boldsymbol{l},1>} \leq \max(-\delta_{\boldsymbol{l}}, 0)$. Hence, $\sum_{l=1}^{1} \delta_{<\boldsymbol{l},l>} \frac{F_{<\boldsymbol{l},l>}}{F_{\boldsymbol{l}}} \leq \max(-\delta_{\boldsymbol{l}}\Omega, 0)$ for any non-increasing non-negative sequence $\{F_{<\boldsymbol{l},l>}\}$ bounded by $\Omega F_{\boldsymbol{l}}$. Now, assume $\sum_{l=1}^{2} \delta_{<\boldsymbol{l},l>} \frac{F_{<\boldsymbol{l},l>}}{F_{\boldsymbol{l}}} \leq \max(-\delta_{\boldsymbol{l}}\Omega, 0)$,..., $\sum_{l=1}^{k} \delta_{<\boldsymbol{l},l>} \frac{F_{<\boldsymbol{l},l>}}{F_{\boldsymbol{l}}} \leq \max(-\delta_{\boldsymbol{l}}\Omega, 0)$ for any non-increasing non-negative sequence $\{F_{<\boldsymbol{l},l>}\}$ bounded by $\Omega F_{\boldsymbol{l}}$. We will show that this implies $\sum_{l=1}^{k+1} \delta_{<\boldsymbol{l},l>} \frac{F_{<\boldsymbol{l},l>}}{F_{\boldsymbol{l}}} \leq \max(-\delta_{\boldsymbol{l}}\Omega, 0)$ for any non-increasing non-negative sequence $\{F_{<\boldsymbol{l},l>}\}$ bounded by $\Omega F_{\boldsymbol{l}}$. There are two cases to consider. If $\delta_{<\boldsymbol{l},k+1>} \leq 0$, then the result immediately follows. On the other hand, if $\delta_{<\boldsymbol{l},k+1>} > 0$, by assumption, we have

$$0 < \delta_{<\boldsymbol{l},k+1>} \leq -\delta_{\boldsymbol{l}} - \sum_{l=1}^{k} \delta_{<\boldsymbol{l},l>}.$$

Hence, substituting $\delta_{<\boldsymbol{l},k+1>}$, we have

$$\begin{aligned}
\sum_{l=1}^{k+1} \delta_{<\boldsymbol{l},l>} \frac{F_{<\boldsymbol{l},l>}}{F_{\boldsymbol{l}}} &\leq \sum_{l=1}^{k} \delta_{<\boldsymbol{l},l>} \frac{F_{<\boldsymbol{l},l>}}{F_{\boldsymbol{l}}} \\
&\quad - \sum_{l=1}^{k} \delta_{<\boldsymbol{l},l>} \frac{F_{<\boldsymbol{l},k+1>}}{F_{\boldsymbol{l}}} - \delta_{\boldsymbol{l}} \frac{F_{<\boldsymbol{l},k+1>}}{F_{\boldsymbol{l}}}. \\
&= \sum_{l=1}^{k} \delta_{<\boldsymbol{l},l>} \left(\frac{F_{<\boldsymbol{l},l>} - F_{<\boldsymbol{l},k+1>}}{F_{\boldsymbol{l}}} \right) \\
&\quad - \delta_{\boldsymbol{l}} \frac{F_{<\boldsymbol{l},k+1>}}{F_{\boldsymbol{l}}}.
\end{aligned}$$

The sequence $F_{<\boldsymbol{l},1>} - F_{<\boldsymbol{l},k+1>}, \ldots, F_{<\boldsymbol{l},k>} - F_{<\boldsymbol{l},k+1>}$ is non-increasing, non-negative and bounded by $(\Omega - \frac{F_{<\boldsymbol{l},k+1>}}{F_{\boldsymbol{l}}})F_{\boldsymbol{l}}$. Hence, by the induction hypothesis, we have

$$\begin{aligned}
&\sum_{l=1}^{k} \delta_{<\boldsymbol{l},l>} \left(\frac{F_{<\boldsymbol{l},l>} - F_{<\boldsymbol{l},k+1>}}{F_{\boldsymbol{l}}} \right) \\
\leq\ &\max\left(-\delta_{\boldsymbol{l}} \left(\Omega - \frac{F_{<\boldsymbol{l},k+1>}}{F_{\boldsymbol{l}}} \right), 0 \right).
\end{aligned}$$

This implies that

$$\begin{aligned}&\sum_{l=1}^{k+1}\delta_{<\boldsymbol{l},l>}\frac{F_{<\boldsymbol{l},l>}}{F_{\boldsymbol{l}}}\\ \leq\ &\max\left(-\delta_{\boldsymbol{l}}\left(\Omega-\frac{F_{<\boldsymbol{l},k+1>}}{F_{\boldsymbol{l}}}\right),0\right)-\delta_{\boldsymbol{l}}\frac{F_{<\boldsymbol{l},k+1>}}{F_{\boldsymbol{l}}}.\end{aligned}$$

Considering the two cases when $\delta_{\boldsymbol{l}}>0$ and $\delta_{\boldsymbol{l}}\leq 0$, we can show that this implies $\sum_{l=1}^{k+1}\delta_{<\boldsymbol{l},l>}\frac{F_{<\boldsymbol{l},l>}}{F_{\boldsymbol{l}}}\leq\max(-\delta_{\boldsymbol{l}}\Omega,0)$. ■

The following Lemma, which uses Lemma 9, is essential to prove Proposition 8.

Lemma 11: If $\boldsymbol{\mu}(t)+\boldsymbol{\delta}$ satisfies the constraints for optA($\boldsymbol{f}(t),\boldsymbol{x}(t)$), then $\boldsymbol{\delta}$ satisfies the following:

(a) Consider any node $\boldsymbol{l}$ that is not a leaf. Let l be any child of $\boldsymbol{l}$. If $<\boldsymbol{l},l>$ is not a leaf and all its children ($i=1,\ldots,C(<\boldsymbol{l},l>)$) satisfy $\delta_{<\boldsymbol{l},l,i>}\leq\max\left(-\delta_{<\boldsymbol{l},l>}-\sum_{j=1}^{i-1}\delta_{<\boldsymbol{l},l,j>},0\right)$, then

$$\delta_{<\boldsymbol{l},l>}\leq\max(-\delta_{\boldsymbol{l}}-\sum_{j=1}^{l-1}\delta_{<\boldsymbol{l},j>},0). \tag{19}$$

(b) Consider any node l that is the child of the root. If l is not a leaf and all its children ($i=1,\ldots,C(l)$) satisfy $\delta_{<l,i>}\leq\max\left(-\delta_l-\sum_{j=1}^{i-1}\delta_{<l,j>},0\right)$, then

$$\delta_l\leq\max(-\sum_{j=1}^{l-1}\delta_j,0). \tag{20}$$

Remark: Part (a) of the Lemma says that if the children of $<\boldsymbol{l},l>$, $<\boldsymbol{l},l,i>$ satisfy the property that, to increase service to $<\boldsymbol{l},l,i>$, we must reduce service from higher priority nodes $<\boldsymbol{l},l>,<\boldsymbol{l},l,1>,\ldots,<\boldsymbol{l},l,i-1>$, then link $<\boldsymbol{l},l>$ also satisfies this property, i.e., to increase service to link $<\boldsymbol{l},l>$, we must reduce service to its higher priority links $\boldsymbol{l},<\boldsymbol{l},1>,\ldots,<\boldsymbol{l},l-1>$. Part (b) is a special case for when the link is directly connected to the root node. The significance of this lemma is that it allows the above mentioned property to propogate up the tree from the leaf nodes. In other words, if a link's children satisfy the property, then the link satisfies the property as well.

Proof: From Lemma 9, we know that

$$\begin{aligned}\delta_{<\boldsymbol{l},l>}\ \leq\ &\max\left(-\delta_{\boldsymbol{l}}-\sum_{j=1}^{l-1}\delta_{<\boldsymbol{l},j>},\right.\\ &\left.\sum_{j=1}^{C(<\boldsymbol{l},l>)}\delta_{<\boldsymbol{l},l,j>}\frac{F_{<\boldsymbol{l},l,j>}}{F_{<\boldsymbol{l},l>}})\right).\end{aligned} \tag{21}$$

By Lemma 10 and the assumptions on links $<\boldsymbol{l},l,i>$, we have,

$$\sum_{i=1}^{C(<\boldsymbol{l},l>)}\delta_{<\boldsymbol{l},l,i>}\frac{F_{<\boldsymbol{l},l>}}{F_{\boldsymbol{l}}}\leq\max(-\delta_{<\boldsymbol{l},l>}\Omega,0).$$

Using this in (21), we obtain

$$\delta_{<\boldsymbol{l},l>}\leq\max\left(-\delta_{\boldsymbol{l}}-\sum_{j=1}^{l-1}\delta_{<\boldsymbol{l},j>},\max(-\delta_{<\boldsymbol{l},l>},0)\Omega\right). \tag{22}$$

Considering the two cases $\delta_{<\boldsymbol{l},l>}>0$ and $\delta_{<\boldsymbol{l},l>}\leq 0$, (22) can be shown to imply (19). The proof of (20) follows a similar idea. ■

As we mentioned before, the leaf nodes satisfy the property that an increase in service to the link must come at a reduction in service to higher priority links (see Lemma 9). Lemma 11 states that if a link's children satisfy this property, then the link itself must also satisfy this property. Clearly, this idea leads to the propogation of this property up the tree from the leaf nodes and hence we expect that all links in the tree must satisfy this property. This result is explicitly stated in the following Lemma.

Lemma 12: If $\boldsymbol{\mu}(t)+\boldsymbol{\delta}$ satisfies the constraint equations for optA($\boldsymbol{f}(t),\boldsymbol{x}(t)$), $\boldsymbol{\delta}$ satisfies the following:

(a) Consider any node $\boldsymbol{l}$ that is not a leaf. Let l be any child of $\boldsymbol{l}$. Then,

$$\delta_{<\boldsymbol{l},l>} \leq \max(-\delta_{\boldsymbol{l}} - \sum_{j=1}^{l-1} \delta_{<\boldsymbol{l},j>}, 0). \tag{23}$$

(b) Consider any node l that is the child of the root. Then,

$$\delta_l \leq \max(-\sum_{j=1}^{l-1} \delta_j, 0). \tag{24}$$

Proof: To prove (a), we first assume that there is atleast one link such that it does not satisfy (23). Let $<\boldsymbol{l},l>$ be such a link with the largest link depth (i.e. the link with the most number of hops from the root). Then, either link $<\boldsymbol{l},l>$ has children and the child nodes satisfy $\delta_{<\boldsymbol{l},l,i>} \leq \max(-\delta_{<\boldsymbol{l},l>} - \sum_{j=1}^{i-1}\delta_{<\boldsymbol{l},l,j>}, 0)$, in which case Lemma 11 part (a) applies and leads to a contradiction, or $<\boldsymbol{l},l>$ is a leaf node in which case Lemma 9 applies leading to a contradiction.

To prove (b), we use the result (a). Assume that link l does not satisfy (24). Either link l has children, in which case by part (a), we know that the children $<l,1>,\ldots,<l,C(l)>$ satisfy

$$\delta_{<l,i>} \leq \max\left(-\delta_l - \sum_{j=1}^{i-1}\delta_{<l,j>}, 0\right).$$

Lemma 11 part (b) then applies and we have a contradiction. The other situation is that link l is a leaf node. In this case Lemma 9 applies and we have a contradiction. ■

We are now ready to prove Proposition 8.

Proof: [of Proposition 8] By Lemma 12, we know that $\delta_l \leq \max(-\sum_{j=1}^{l-1}\delta_j, 0)$ for $l=1,\ldots,C$.

To prove the proposition, it is enough to show that for any mathematical quantities δ_l which satisfy the inequalities

$$\delta_l \leq \max(-\sum_{j=1}^{l-1}\delta_j, 0), \tag{25}$$

for $l=1,\ldots,C$, and any non-increasing, non-negative sequence $\{F_l\}$, the following is true $\sum_{l=1}^{C}\delta_l F_l \leq 0$.

We emphasize that this is a purely mathematical result and that allowing $\{F_l\}$ to represent various sequences is simply a trick to shorten the proof. It does not mean that we consider various systems with different values for the link capacities.

We will prove this by induction. By (25), we know that $\delta_1 \leq 0$. Hence, $\sum_{l=1}^{1}\delta_l F_l \leq 0$ for any non-increasing, non-negative sequence of numbers $\{F_l\}$. Now, assume $\sum_{l=1}^{2}\delta_l F_l \leq 0$,...,$\sum_{l=1}^{k-1}\delta_l F_l \leq 0$,$\sum_{l=1}^{k}\delta_l F_l \leq 0$ for any non-increasing, non-negative sequence of numbers $\{F_l\}$. We will show that this implies $\sum_{l=1}^{k+1}\delta_l F_l \leq 0$. There are two cases to consider. If $\delta_{k+1} \leq 0$, then the result immediately follows. On the other hand, if $\delta_{k+1} > 0$, by (25), we have $0 < \delta_{k+1} \leq -\sum_{l=1}^{k}\delta_l$. Hence, $\sum_{l=1}^{k+1}\delta_l F_l \leq \sum_{l=1}^{k}\delta_l F_l - \sum_{l=1}^{k}\delta_l F_{k+1} = \sum_{l=1}^{k}\delta_l\left(F_l - F_{k+1}\right)$.

Since $\{F_l\}$ is a non-increasing sequence, the sequence $F_1 - F_{k+1},\ldots,F_k - F_{k+1}$ is a non-increasing non-negative sequence. Hence, by the induction hypothesis, $\sum_{l=1}^{k}\delta_l\left(F_l - F_{k+1}\right) < 0$. This implies $\sum_{l=1}^{k+1}\delta_l F_l \leq 0$ for any non-increasing, non-negative sequence $\{F_l\}$. ■

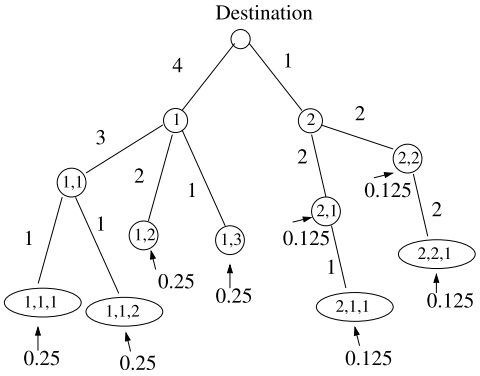

Fig. 1. System topology for simulation

VI. Simulation

In this section, we present simulation results for the topology shown in figure 1. Note that the nodes/links are labelled according to the scheme in section II. This topology consists of 12 nodes with two nodes at depth 1, 5 nodes at depth 2 and 4 nodes at depth 3. Six of the nodes are leaf nodes. There are 8 flows in the network, each with the root as the destination. In each time slot, one packet arrives at (or is generated by) each source node with a certain fixed probability, independent of other flows and other time slots. The average arrival rate for the flow originating at a node is labelled on the node. For example, the average arrival rate for the flow originating at node $(1,1,1)$ is 0.25. The numbers near the links denote the capacity of the link. For example, link $(1,1)$ has capacity 3 and link $(2,1)$ has capacity 2. We define $S(\boldsymbol{X}(t)) = X_1(t)+X_2(t)+X_{(1,1)}(t)+X_{(1,2)}(t)+X_{(1,3)}(t)+X_{(2,1)}(t)+X_{(2,2)}(t)+X_{(1,1,1)}(t)+X_{(1,1,2)}(t)+X_{(2,1,1)}(t)+X_{(2,2,1)}(t)$.

Our metric of interest is the overflow probability $\mathbf{P}[S(\boldsymbol{X}(t)) > B]$. We simulate the system under different scheduling policies: P-TREE scheduler, back-pressure & back-pressure-α schedulers and the multi-hop version of greedy maximal matching (GMM).

Let us briefly review the back-pressure [1] and greedy maximal matching [20] policies. Both policies have the following common features. The differential backlog across a link is the difference of the backlog at the source node of the link and that at the destination node of the link. For example, the differential backlog of the link $(1,1)$ is $X_{(1,1)} - X_1$. Each link is assigned a weight $W_{\boldsymbol{l}}$ that is the product of the differential backlog and the link capacity. For example, $W_{(1,1)} = (X_{(1,1)} - X_1)3$. The back-pressure scheduler will activate links (subject to interference constraints) in such a fashion as to maximize the sum of the weights of the activated links. The greedy maximal matching will instead do the following. It will first activate the link with the largest weight. Then, it will remove from consideration all links that interfere with this activated link. From the remaining links, it will activate the link with the largest weight and remove from consideration the links that interfere with this link. This procedure is repeated till there are no more links available.

The back-pressure-α algorithm is similar to the back-pressure algorithm except that instead of taking the difference of the backlogs, the algorithm takes the difference of the backlogs raised to a power α. That is, the weight of link $(1,1)$ will be $W_{(1,1)} = (X_{(1,1)}^{\alpha} - X_1^{\alpha})3$. It can be shown that this algorithm minimizes the drift of the lyapunov function $(\sum_{\boldsymbol{l}\in\mathcal{L}} X_{\boldsymbol{l}}^{\alpha+1})^{1/(\alpha+1)}$ and hence it is large deviations decay-rate optimal for the probability of overflow $\mathbf{P}((\sum_{\boldsymbol{l}\in\mathcal{L}} X_{\boldsymbol{l}}^{\alpha+1})^{1/(\alpha+1)} > B)$ [13]. As $\alpha \to 0$, we have $(\sum_{\boldsymbol{l}\in\mathcal{L}} X_{\boldsymbol{l}}^{\alpha+1})^{1/(\alpha+1)} \to \sum_{\boldsymbol{l}\in\mathcal{L}} X_{\boldsymbol{l}}$. Hence, as $\alpha \to 0$, one would expect this algorithm to have near-optimal performance in terms of the decay-rate for $\mathbf{P}(\sum_{\boldsymbol{l}\in\mathcal{L}} X_{\boldsymbol{l}} > B)$.

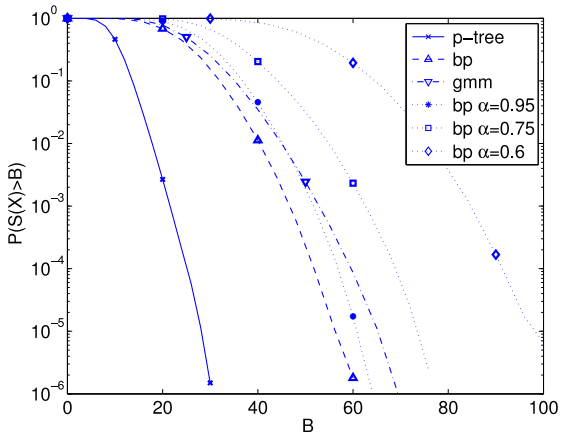

Fig. 2. The overflow probability of the sum-queue versue threshold B.

One of the problems with the back-pressure & back-pressure-α schedulers is that they entail a high computational complexity due to the fact that the algorithms have to search for the best way to activate links in order to maximize the total weight. The greedy maximal matching algorithm overcomes this issue [20], [26]. For the node-exclusive interference model that we consider, the back-pressure and back-pressure-α schedulers reduces to a matching problem which has complexity $O(|\mathcal{L}|^3)$ [19]. The greedy maximal matching algorithm has complexity $O(|\mathcal{L}|\log(|\mathcal{L}|))$ [20]. Our proposed P-TREE algorithm has an even lower complexity of $O(|\mathcal{L}|)$.

In figure 2, we plot $\mathbf{P}[S(X) > B]$ vs B with the y-axis in log scale. We observe that the P-TREE scheduler has the best decay rate and indeed performs much better than the other schedulers. The back-pressure-α algorithm appears to perform very poorly as α is reduced. This is because of the large-deviations decay-rate kicking in at higher and higher values of the threshold B. This effect has been documented in detail in our other works [6], [27]. In contrast, our P-TREE algorithm not only maximizes the decay rate but also performs very well when overflow thresholds are small.

VII. Conclusion

In this work, we consider the problem of scheduling links in a wireless multi-hop system performing convergecast. The goal of the scheduling algorithm is to minimize the sum-queue backlog over the network. We design a novel low complexity scheduling algorithm called P-TREE scheduler and prove that this scheduler maximizes the decay rate of the probability that the sum-queue exceeds a certain threshold. We use simulations to compare this algorithm with the well known back-pressure scheduler and the multi-hop version of greedy maximal matching scheduler. The P-TREE scheduler is seen to perform much better than these well known algorithms not only in terms of decay rate but also in terms of actual probabilities of overflow at small overflow thresholds.

Appendix

Proof of Lemma 4

In the following, we take $||\cdot||$ to be the L_1 norm.

We restate the assumptions from [13] for reference.

A. Restatement of assumptions from [13]

Assumption 1: The Lyapunov function $V(\boldsymbol{x})$, defined for $\boldsymbol{x} \geq 0$, satisfies the following:

1) $V(\boldsymbol{x})$ is a continuous function of $\boldsymbol{x}$.
2) $V(\boldsymbol{x}) \geq 0$ for all $\boldsymbol{x}$ and $V(\boldsymbol{x}) = 0$ if and only if $\boldsymbol{x} = 0$.
3) $V(\boldsymbol{x}) \to \infty$ if $||\boldsymbol{x}|| \to \infty$.
4) $\min_{||\boldsymbol{x}||\geq 1} V(\boldsymbol{x}) \geq 1$. Further there exists a number $\tilde{C}$ such that $\max_{||\boldsymbol{x}||\leq 1} V(\boldsymbol{x}) \leq \tilde{C}$.
5) For any $\mathcal{B} > 0$, there exists a constant $\mathcal{L}$ that may depend on $\mathcal{B}$, such that for any $||\boldsymbol{x}_1|| \leq \mathcal{B}$ and $||\boldsymbol{x}_2|| \leq \mathcal{B}$,
$$|V(\boldsymbol{x}_1) - V(\boldsymbol{x}_2)| \leq \mathcal{L}||\boldsymbol{x}_1 - \boldsymbol{x}_2||.$$
6) The following holds (for a fixed arrival rate $\hat{\boldsymbol{\lambda}}$ assumed in the system model): For all fluid limits $\boldsymbol{x}(t)$ (i.e. fluid sample path with $\boldsymbol{f}(t) = \hat{\boldsymbol{\lambda}}$ for all t), when $V(\boldsymbol{x}(t)) > 0$,
$$\frac{d}{dt}V(\boldsymbol{x}(t)) \leq -\eta, \tag{26}$$
for almost all t, where η is a positive constant.

Parts (1)-(3) and (6) of the assumption are typically used when establishing stability through Lyapunov functions. Part (6) states that the Lyapunov function must have negative drift when the arrival process does not deviate from its mean behavior. This implies stability of the system since the negative drift will prevent the Lyapunov function from becoming exceedingly large.

Assumption 2:
1) There exists $\epsilon > 0$ such that for all fluid sample paths and for all time t with $||\boldsymbol{f}(t) - \hat{\boldsymbol{\lambda}}|| \leq \epsilon$ and $V(\boldsymbol{x}(t)) > 0$, the following holds:
$$\frac{d}{dt}V(\boldsymbol{x}(t)) \leq -\frac{\eta}{2},$$
where $\eta > 0$ is the same constant as in (26).
2) For any $\delta > 0$, there exists $M_1 \geq 0$ such that for all fluid sample paths and for all time t with $||\boldsymbol{f}(t) - \hat{\boldsymbol{\lambda}}|| \geq \delta$, the following holds,
$$\frac{d}{dt}V(\boldsymbol{x}(t)) \leq M_1.$$

Part (1) of this assumption states that if the arrival process deviates from the mean behaviour slightly, the Lyapunov function still experiences negative drift leading to system stability. Part (2) states that even if the arrival process deviates significantly from its mean behaviour, the rate of growth of the Lyapunov function is still bounded.

Assumption 3: The Lyapunov function $V(\cdot)$ is linear in scale, i.e., $V(c\boldsymbol{x}) = cV(\boldsymbol{x})$ for all $c \geq 0$.

Assumption 5: $V(\boldsymbol{x})$ is non-decreasing in each component x_l.

Assumption 6: $V(\boldsymbol{x}_1 + \boldsymbol{x}_2) \leq V(\boldsymbol{x}_1) + V(\boldsymbol{x}_2)$ for any two vectors $\boldsymbol{x}_1 \geq 0$ and $\boldsymbol{x}_2 \geq 0$,

Assumptions 3 and 6 combined imply that Lyapunov function $V(\cdot)$ behaves almost like a norm except that it may not be defined when components of $\boldsymbol{x}$ are negative.

All the assumptions other than assumption 1 part 6) and assumption 2 are easy to verify. We do not provide details for them here. In what follows, we will show that assumption 1 part 6) and assumption 2 are true.

First, we verify assumption 1 part 6). We need to show that the drift of the Lyapunov function $V(\boldsymbol{x}(t))$, when the arrival rate is $\boldsymbol{f}(t) = \hat{\boldsymbol{\lambda}}$, is less than $-\eta$ for some $\eta > 0$ when the Lyapunov function $V(\boldsymbol{x}(t)) > 0$. If $V(\boldsymbol{x}(t)) > 0$, there must exist

some queue $\hat{\boldsymbol{l}}$ with $x_{\hat{\boldsymbol{l}}}(t) > 0$. Since $\hat{\boldsymbol{\lambda}}$ is in the stability region of the system, there exists $\eta > 0$ such that $\hat{\boldsymbol{\lambda}} + \mathbf{1}\eta$, where $\mathbf{1}\eta$ is the vector with all entries equal to η, is in the capacity region. This means that the system can be stabilized when the arrival rate for node $\boldsymbol{l}$ is $\hat{\lambda}_{\boldsymbol{l}}$ for $\boldsymbol{l} \neq \hat{\boldsymbol{l}}$ and the arrival rate for $\hat{\boldsymbol{l}}$ is $\hat{\lambda}_{\hat{\boldsymbol{l}}} + \eta$. The reason we only add η to $\hat{\boldsymbol{l}}$ is because since $x_{\hat{\boldsymbol{l}}}(t) > 0$, the flow constraint $\mu_{\hat{\boldsymbol{l}}}(t) \leq \lambda_{\hat{\boldsymbol{l}}}(t) + \sum_{l=1}^{C(\hat{\boldsymbol{l}})} \frac{F_{<\hat{\boldsymbol{l}},l>}}{F_{\hat{\boldsymbol{l}}}} \mu_{<\hat{\boldsymbol{l}},l>}$ does not appear in optA($\hat{\boldsymbol{\lambda}}, \boldsymbol{x}(t)$). This property will be necessary to show negative drift. Since the system can be stabilized, there exists constant $\hat{\boldsymbol{\mu}}$ such that

$$\begin{aligned}
&\sum_{\boldsymbol{l} \in \mathcal{L}} F_{\boldsymbol{l}} \hat{\lambda}_{\boldsymbol{l}} + \eta - \sum_{l=1}^{C} F_l \hat{\mu}_l \leq 0 \\
&\sum_{l=1}^{C} \hat{\mu}_l \leq 1 \\
&\sum_{l=1}^{C(\boldsymbol{l})} \hat{\mu}_{<\boldsymbol{l},l>} + \hat{\mu}_{\boldsymbol{l}} \leq 1 \text{ for all interior nodes } \boldsymbol{l}. \\
&\hat{\mu}_{\boldsymbol{l}}(t) \in [0,1] \text{ for all nodes } \boldsymbol{l}.
\end{aligned} \tag{27}$$

For all links $\boldsymbol{l} \neq \hat{\boldsymbol{l}}$:

$$\hat{\mu}_{\boldsymbol{l}} = \hat{\lambda}_{\boldsymbol{l}} + \sum_{l=1}^{C(\boldsymbol{l})} \frac{F_{<\boldsymbol{l},l>}}{F_{\boldsymbol{l}}} \hat{\mu}_{<\boldsymbol{l},l>} \text{ if } \boldsymbol{l} \text{ is an interior node.}$$

$$\hat{\mu}_{\boldsymbol{l}} = \hat{\lambda}_{\boldsymbol{l}} \text{ if } \boldsymbol{l} \text{ is a leaf.}$$

$$\hat{\mu}_{\hat{\boldsymbol{l}}} = \hat{\lambda}_{\hat{\boldsymbol{l}}} + \eta + \sum_{l=1}^{C(\hat{\boldsymbol{l}})} \frac{F_{<\boldsymbol{l},l>}}{F_{\boldsymbol{l}}} \hat{\mu}_{<\boldsymbol{l},l>} \text{ if } \hat{\boldsymbol{l}} \text{ is an interior node.}$$

$$\hat{\mu}_{\hat{\boldsymbol{l}}} = \hat{\lambda}_{\hat{\boldsymbol{l}}} + \eta \text{ if } \hat{\boldsymbol{l}} \text{ is a leaf.}$$

One can think of $\hat{\mu}_{\boldsymbol{l}}$ as the long term service rate for link $\boldsymbol{l}$ provided by an algorithm that stabilizes the system.

Clearly $\hat{\boldsymbol{\mu}}$ satisfies the constraint equations of optA($\hat{\boldsymbol{\lambda}}, \boldsymbol{x}(t)$). Hence we have optA($\hat{\boldsymbol{\lambda}}, \boldsymbol{x}(t)$)$\leq \sum_{\boldsymbol{l} \in \mathcal{L}} F_{\boldsymbol{l}} \hat{\lambda}_{\boldsymbol{l}} - \sum_{l=1}^{C} F_l \hat{\mu}_l$. From (27), we then have optA($\hat{\boldsymbol{\lambda}}, \boldsymbol{x}(t)$)$\leq -\eta$. From Lemma 5, we know that the drift of the Lyapunov function for the p-tree algorithm is given by optA($\hat{\boldsymbol{\lambda}}, \boldsymbol{x}(t)$). Hence, we have proved assumption 1 part 6).

We can prove assumption 2 in a similar manner. Since $\hat{\boldsymbol{\lambda}}$ is in the capacity region of the system, there exists an $\epsilon < \eta/2$ such that both $\boldsymbol{f}(t)$ and $\boldsymbol{f}(t) + \mathbf{1}\eta/2$ are in the capacity region whenever $||\boldsymbol{f}(t) - \hat{\boldsymbol{\lambda}}|| < \epsilon$. Again, as before, let $\hat{\boldsymbol{l}}$ be a link with $x_{\hat{\boldsymbol{l}}}(t) > 0$. This means that we can find constants $\hat{\boldsymbol{\mu}}$ such that

$$\begin{aligned}
&\sum_{\boldsymbol{l} \in \mathcal{L}} F_{\boldsymbol{l}} f_{\boldsymbol{l}}(t) + \eta/2 - \sum_{l=1}^{C} F_l \hat{\mu}_l \leq 0 \\
&\sum_{l=1}^{C} \hat{\mu}_l \leq 1 \\
&\sum_{l=1}^{C(\boldsymbol{l})} \hat{\mu}_{<\boldsymbol{l},l>} + \hat{\mu}_{\boldsymbol{l}} \leq 1 \text{ for all interior nodes } \boldsymbol{l}. \\
&\hat{\mu}_{\boldsymbol{l}}(t) \in [0,1] \text{ for all nodes } \boldsymbol{l}.
\end{aligned} \tag{28}$$

For all links $\boldsymbol{l} \neq \hat{\boldsymbol{l}}$:

$$\hat{\mu}_{\boldsymbol{l}} = f_{\boldsymbol{l}}(t) + \sum_{l=1}^{C(\boldsymbol{l})} \frac{F_{<\boldsymbol{l},l>}}{F_{\boldsymbol{l}}} \hat{\mu}_{<\boldsymbol{l},l>} \text{ if } \boldsymbol{l} \text{ is an interior node.}$$

$$\hat{\mu}_{\boldsymbol{l}} = f_{\boldsymbol{l}}(t) \text{ if } \boldsymbol{l} \text{ is a leaf.}$$

$$\hat{\mu}_{\hat{\boldsymbol{l}}} = f_{\hat{\boldsymbol{l}}}(t) + \eta/2 + \sum_{l=1}^{C(\hat{\boldsymbol{l}})} \frac{F_{<\boldsymbol{l},l>}}{F_{\boldsymbol{l}}} \hat{\mu}_{<\boldsymbol{l},l>} \text{ if } \hat{\boldsymbol{l}} \text{ is an interior node.}$$

$$\hat{\mu}_{\hat{\boldsymbol{l}}} = f_{\hat{\boldsymbol{l}}}(t) + \eta/2 \text{ if } \hat{\boldsymbol{l}} \text{ is a leaf.}$$

$\hat{\boldsymbol{\mu}}$ satisfies the constraint equations of optA($\boldsymbol{f}(t), \boldsymbol{x}(t)$) which implies that optA($\boldsymbol{f}(t), \boldsymbol{x}(t)$)$\leq \sum_{\boldsymbol{l} \in \mathcal{L}} F_{\boldsymbol{l}} f_{\boldsymbol{l}}(t) - \sum_{l=1}^{C} F_l \hat{\mu}_l$. From (28), we then have optA($\boldsymbol{f}(t), \boldsymbol{x}(t)$)$\leq -\eta/2$. From Lemma 5, we know that the drift of the Lyapunov function for the p-tree algorithm is given by optA($\boldsymbol{f}(t), \boldsymbol{x}(t)$). Hence, we have proved assumption 2 part 1). Assumption 2 part 2) holds because optA($\boldsymbol{f}(t), \boldsymbol{x}(t)$) is bounded from above by $\sum_{\boldsymbol{l} \in \mathcal{L}} M$ where M is the bound on $A_{\boldsymbol{l}}(t)$.

REFERENCES

[1] L. Tassiulas and A. Ephremides, "Stability Properties of Constrained Queueing Systems and Scheduling Policies for Maximum Throughput in Multihop Radio Networks," *IEEE Transactions on Automatic Control*, vol. 37, no. 12, pp. 1936–1948, December 1992.

[2] M. J. Neely, E. Modiano, and C. E. Rohrs, "Dynamic Power Allocation and Routing for Time Varying Wireless Networks," *IEEE Journal on Selected Areas in Communications, Special Issue on Wireless Ad-Hoc Networks*, vol. 23, no. 1, pp. 89–103, January 2005.

[3] S. Shakkottai, R. Srikant, and A. Stolyar, "Pathwise Optimality of the Exponential Scheduling Rule for Wireless Channels," *Advances in Applied Probability*, pp. 1021–1045, December 2004.

[4] B. Sadiq, S. J. Baek, and G. de Veciana, "Delay-Optimal Opportunistic Scheduling and Approximations: The Log Rule." in *Proceedings of IEEE INFOCOM*, April 2009.

[5] X. Lin and V. J. Venkataramanan, "On the Large-Deviations Optimality of Scheduling Policies Minimizing the Drift of a Lyapunov Function," in *47th Annual Allerton Conference on Communication, Control, and Computing*, Monticello, IL, September 2009.

[6] V. J. Venkataramanan and X. Lin, "On Wireless Scheduling Algorithms for Minimizing the Queue-Overflow Probability," *IEEE/ACM Trans. on Networking*, vol. 18, no. 3, June 2010.

[7] L. Ying, S. Shakkottai, and A. Reddy, "On Combining Shortest-Path and Back-Pressure Routing Over Multihop Wireless Networks," in *Proceedings of IEEE INFOCOM*, April 2009.

[8] L. Bui, R. Srikant, and A. L. Stolyar, "Novel Architectures and Algorithms for Delay Reduction in Back-Pressure Scheduling and Routing," in *Infocom Mini-Conference*, April 2009.

[9] M. J. Neely, "Order Optimal Delay for Opportunistic Scheduling in Multi-User Wireless Uplinks and Downlinks," *IEEE/ACM Transactions on Networking*, 2008.

[10] ——, "Delay Analysis for Maximal Scheduling in Wireless Networks with Bursty Traffic," in *Proceedings of IEEE INFOCOM*, April 2008.

[11] A. L. Stolyar, "MaxWeight Scheduling in a Generalized Switch: State Space Collapse and Workload Minimization in Heavy Traffic," *Annals of Applied Probability*, vol. 14, no. 1, pp. 1–53, 2004.

[12] S. Shakkottai, "Effective Capacity and QoS for Wireless Scheduling," *IEEE Transactions on Automatic Control*, vol. 53, no. 3, April 2008.

[13] V. J. Venkataramanan and X. Lin, "On the Queue-Overflow Probability of Wireless Systems: A New Approach Combining Large Deviations with Lyapunov Functions," *submitted to IEEE Trans. on Information Theory*, 2009. [Online]. Available: http://min.ecn.purdue.edu/%7Elinx/publications.html

[14] A. L. Stolyar and K. Ramanan, "Largest Weighted Delay First Scheduling: Large Deviations and Optimality," *Annals of Applied Probability*, vol. 11, no. 1, pp. 1–48, 2001.

[15] A. L. Stolyar, "Large Deviations of Queues Sharing a Randomly Time-varying Server," *Queueing Systems*, vol. 59, pp. 1–35, 2008.

[16] G. R. Gupta and N. B. Shroff, "Delay Analysis for Multi-hop Wireless Networks," in *Proceedings of IEEE INFOCOM*, April 2009.

[17] S. Jagabathula and D. Shah, "Optimal Delay Scheduling in Networks with Arbitrary Constraints," in *ACM SIGMETRICS/Performance*, June 2008.

[18] L. Tassiulas and A. Ephremides, “Dynamic Scheduling for Minimum Delay in Tandem and Parallel Constrained Queueing Models,” *Annals of Operation Research*, vol. 48, pp. 333–355, 1994.

[19] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*. Englewood Cliffs, New Jersey: Prentice-Hall, 1982.

[20] X. Lin and N. B. Shroff, “The Impact of Imperfect Scheduling on Cross-Layer Congestion Control in Wireless Networks,” *IEEE/ACM Transactions on Networking*, vol. 14, no. 2, pp. 302–315, 2006.

[21] C. Joo, X. Lin, and N. B. Shroff, “Greedy Maximal Matching: Performance Limits for Arbitrary Network Graphs Under the Node-exclusive Interference Model,” *To appear in IEEE Trans. on Automatic Control*.

[22] Y. Yi and S. Shakkottai, “Hop-by-hop Congestion Control over a Wireless Multi-hop Network,” in *Proceedings of IEEE INFOCOM*, March 2004.

[23] S. Sarkar and L. Tassiulas, “End-to-end Bandwidth Guarantees Through Fair Local Spectrum Share in Wireless Ad-hoc Networks,” in *Proceedings of the IEEE Conference on Decision and Control*, December 2003.

[24] A. Dembo and O. Zeitouni, *Large Deviations Techniques and Applications*, 2nd ed. New York: Springer-Verlag, 1998.

[25] S. P. Meyn, “Stability and Asymptotic Optimality of Generalized MaxWeight Policies,” *SIAM J. Control and Optimization*, vol. 47, no. 6, 2009.

[26] C. Joo, X. Lin, and N. B. Shroff, “Understanding the Capacity Region of the Greedy Maximal Scheduling Algorithm in Multi-hop Wireless Networks,” in *Proceedings of IEEE INFOCOM*, April 2008.

[27] V. J. Venkataramanan, X. Lin, L. Ying, and S. Shakkottai, “On Scheduling for Minimizing End-to-End Buffer Usage over Multihop Wireless Networks,” in *Proceedings of IEEE INFOCOM*, March 2010.