Hop-by-Hop Congestion Control Over a Wireless Multi-Hop Network

Yung Yi, *Member, IEEE*, and Sanjay Shakkottai, *Member, IEEE*

***Abstract*—This paper focuses on congestion control over multi-hop, wireless networks. In a wireless network, an important constraint that arises is that due to the MAC (Media Access Control) layer. Many wireless MACs use a time-division strategy for channel access, where, at any point in space, the physical channel can be accessed by a single user at each instant of time.**

In this paper, we develop a fair hop-by-hop congestion control algorithm with the MAC constraint being imposed in the form of a channel access time constraint, using an optimization-based framework. In the absence of delay, we show that this algorithm are globally stable using a Lyapunov-function-based approach. Next, in the presence of delay, we show that the hop-by-hop control algorithm has the property of spatial spreading. In other words, focused loads at a particular spatial location in the network get "smoothed" over space. We derive bounds on the "peak load" at a node, both with hop-by-hop control, as well as with end-to-end control, show that significant gains are to be had with the hop-by-hop scheme, and validate the analytical results with simulation.

***Index Terms*—Control theory, mathematical programming/ optimization.**

I. Introduction

WE CONSIDER the problem of congestion control over wireless, multi-hop networks. Nodes in such networks are radio-equipped, and communicate by broadcasting over wireless links. Communication paths between nodes which are not in radio range of each other are established by intermediate nodes acting as relays to forward data toward the destination. The diverse applications of such networks range from community-based roof-top networks to large-scale ad hoc networks. Over the past few years, the problem of congestion control has received wide-spread attention in the Internet context, where most of this research has focused on modeling, analysis, algorithm development of end-to-end control schemes (such as TCP), and adaptation of such schemes to ad hoc networks [1]. Recent work on congestion control problem provides an optimization-based framework for Internet congestion control and derives a differential-equation-based distributed solution in presence or absence of feedback delay with both primal [2]–[9] and dual [10], [11] approach. Given routing path and bandwidth constraints, algorithms have been developed which converge and have a stable operation.

Manuscript received May 6, 2004; revised December 23, 2005, and January 31, 2006; approved by IEEE/ACM Transactions on Networking Editor R. Mazumdar. This work was supported by NSF Grants ACI-0305644, CNS-0325788, CNS-0347400, and a grant from the Texas Telecommunications Engineering Program (TxTEC). A shorter version of this paper appeared in the Proceedings of IEEE INFOCOM, Hong Kong, March 2004.

Y. Yi was with the Department of Electrical and Computer Engineering, University of Texas at Austin, Austin, TX 78712-0240 USA. He is now with the Department of Electrical Engineering, Princeton University, Princeton, NJ 08544 USA (e-mail: yyi@princeton.edu).

S. Shakkottai is with the Department of Electrical and Computer Engineering, University of Texas at Austin, Austin, TX 78712-0240 USA (e-mail: shakkott@ece.utexas.edu).

Digital Object Identifier 10.1109/TNET.2006.890121

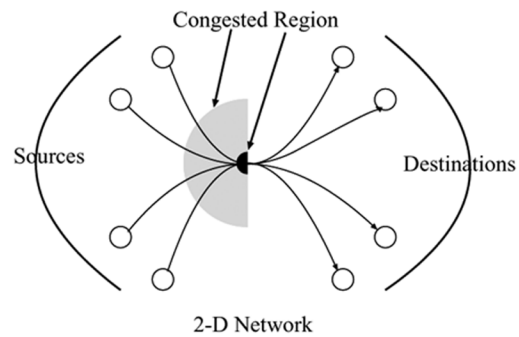


Fig. 1. Spatial Spreading with hop-by-hop controllers.

In a wireless context, however, an important additional resource constraint that arises is that due to the Media Access Control (MAC). To address this, we consider a wireless system, where multiple frequencies/codes are available for transmission, and enables parallel communication in a neighborhood using such orthogonal channels. The wireless MAC in such a system use a time-division strategy for channel access [12], [13], where, at any point in space, *the physical channel can be accessed by a single user at each instant of time* (see Section II-A for details).

This paper formulates an optimization framework for congestion control algorithm in wireless multi-hop networks with the constraint imposed by the MAC. We develop a distributed, *hop-by-hop* congestion control scheme, which is shown to be stable in the absence of propagation delays, and allocates bandwidth to various users in a proportionally fair manner. In the presence of delay, we show that it has the property of *spatial spreading*. In other words, focused loads at a particular spatial location in the network get "smoothed" over space. In Fig. 1, we illustrate this effect. Consider a node accessed by a number of flows. While an end-to-end control scheme could result in large transient overloads (due to delayed feedback) at a single node, a hop-by-hop scheme will "push-back" and cause congestion to occur over space, resulting in smaller peak overloads. Thus, even if the bottleneck node is very close to the receiver (the "worst case" for a hop-by-hop scheme), there are potential gains to be had due to spatial spreading. *Hence, even if the total buffer requirement over the network is the same, the hop-by-hop scheme ensures that the buffers required are spatially spread.*

Hop-by-hop congestion control algorithms have been studied in the wireline context [14]–[17]. Such schemes provide feedback about the congestion state at a node to the hop preceding

it. The preceding node then adapts its transmission-rate-based on this feedback. Feedback is typically provided based on the queue length at the congested node. If the queue length exceeds a threshold, congestion is indicated and the preceding node is notified in order to decrease its transmission rate. It is well known that such schemes, by reacting to congestion faster than end-to-end schemes (the bottleneck node would send feedback *backward*, thus decreasing the delay in the control loop), result in better performance than a corresponding end-to-end scheme. However, Internet congestion control has been dominated by end-to-end schemes (in particular, TCP), and congestion control research in the recent past has focused on the end-to-end schemes, primarily due to scalability and deployability. Hop-by-hop schemes require to have per-flow state management in intermediate nodes, which generates scalability problems.

However, in a wireless network, the number of flows per node is of a much smaller order than in the Internet. Further, wireless networks usually have per-flow queueing for reasons of packet scheduling [12], [18], [19], and the fact that different users are at different locations, thus requiring different physical layer strategies (such as the channel coding and modulation scheme of the power level). In fact, recent studies indicate that per-flow handling may be feasible even in the Internet. In the Internet context, the authors in [20] have measurements to suggest that even in the Internet, per-flow queueing is possible, as the number of *active* flows is in the tens or few hundreds, which can be supported. Thus, the hop-by-hop schemes seem feasible over a wireless multi-hop network.

Related work includes [21], where the authors consider max-min fair scheduling in the context of a wireless network using a similar model as that considered here for media access control (MAC). The authors develop a token-based local scheduling policy at each node to ensure max-min fairness. The work on congestion (or rate) control algorithm in wireless ad hoc networks also has been considered in the context of cross-layer design, where medium access control was jointly studied [22], [23]. In [22], the authors focus on proposing a unified framework for joint rate control and medium access scheduling using a dual approach, and proving stability the proposed algorithm in absence of delay. Recent work in [23] generalizes the resource constraint discussed in this paper (by considering secondary contention as well as primary contention) imposed on MAC layer using flow contention graph, and proposes joint congestion control and media scheduling algorithm based on optimization framework in the context of only end-to-end controllers.

This paper differs from the above-mentioned work in that under MAC constraints in (multi-channel) wireless ad hoc networks, we develop rate-based hop-by-hop as well as end-to-end congestion control algorithm with the objective of (weighted) proportionally fair resource allocation among users. In particular, we prove that the proposed hop-by-hop algorithm is also stable in spite of coupling of transmission rates at each hop in absence of delay. Further, we derive bounds on peak queue lengths in the presence of propagation delay, both with an end-to-end and hop-by-hop scheme, and quantitatively demonstrate spatial spreading in the hop-by-hop control.

A. Main Contributions and Organization

The main contributions of this paper are the following.

1) We develop (weighted) proportionally fair congestion control algorithms (both hop-by-hop as well as end-to-end) with the MAC constraint being imposed in the form of a channel access time constraint, using an optimization-based framework. In the absence of delay, we show that these algorithms are globally stable using a Lyapunov-function-based approach. In particular, with the hop-by-hop algorithm, we have a collection of *coupled* controllers due to the fact that each node along a session's path implements a separate congestion controller. We show that this system of coupled controllers converges to a unique fixed point, which satisfies the proportional fairness condition.
2) We consider the evolution of these algorithms in the presence of propagation delay. We analytically show the effect of spatial spreading, by explicitly deriving the reduction in peak buffer overload under the hop-by-hop scheme for a feed-forward network. We show that at a bottleneck node, the difference in the peak queue length between an end-to-end scheme and a hop-by-hop scheme is at least of order $L^{\alpha}N$, where L is the number of hops, N is the number of sessions, and for all $0 < \alpha < 1$.

We begin with a description of the system model in Section II, and discuss a utility-function-based network optimization framework. In Sections III and IV, we develop a distributed end-to-end and hop-by-hop congestion control algorithm, and prove the stability of both algorithms in absence of delay. Next, in Section V, we also develop a distributed end-to-end and hop-by-hop algorithm with delay, based on which, in Section VI we illustrate spatial spreading in a hop-by-hop algorithm by means of deriving bounds on the peak queue lengths in the presence of feedback delay. We provide simulation results in Section VII to validate the analysis.

II. System Model

A. Access Structure and Network Model

Consider a network with a set $\mathbf{L}$ of links, a set $\mathbf{V}$ of vertices (nodes), and let c_l be the finite capacity of link l, for $l \in \mathbf{L}$. Each vertex corresponds to a node in the network. Each data flow r in the network corresponds to an ordered sequence of links $l \in \mathbf{L}$, and we denote $\mathbf{R}$ as the set of possible sessions.[1] Thus, we model a wireless link between any two nodes in the network to have a finite positive capacity. We further assume that there is no data loss due to channel errors, and we ignore the effect of control data on the usage of channels, and implicitly assume no power control at the node.

In reality, wireless channels are time-varying [24], where each link has some average capacity which will depend on the physical layer scheme. However, in this paper, we model the link to have a fixed capacity. Such a modeling is reasonable when the congestion controller changes *slowly* compared to a MAC packet transmission time, and MAC packet transmission spans *multiple channel states,* thus resulting in the channel

[1]We use the words "session" and "flow" interchangeably throughout this paper.

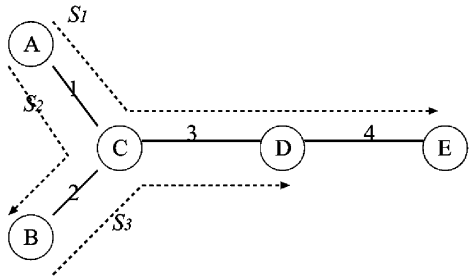


Fig. 2. Example network for time and link constraint.

capacity appearing constant (see also [21], [23] for a similar constant capacity model).

We next describe the access structure considered in this paper. We consider a wireless system, where multiple frequencies/codes are available for transmission (using FDMA/CDMA), where frequency/code (resource) is allowed to be reused in the network, as long as the nodes using the same resource are sufficiently apart and are not involved in the same "contention," enabling parallel communications in a neighborhood using such orthogonal FDMA/CDMA channels (see [21], [25], [26] for additional discussion).

In this wireless system, a single transmission is intended for only one receiver, and each node has only a single transceiver, and hence only half-duplex communication is allowed. Further, a node can successfully receive from at most one other node at the same time. *Thus, at any instant of time, data flows that do not share nodes can transmit/receive simultaneously, but data flows that share a node cannot do so.* In other words, simultaneous transmissions can take place over links (i.e., between a pair nodes) as long as the links do not share a common node. We next describe the constraints on the data flows that follows from this wireless system model.

B. Time Constraint

There are two types of constraints that are imposed, namely, 1) the link constraint and 2) the time constraint. The *link constraint* (usually considered in wired networks) corresponds to the fact that the sum of date rates of all sessions that traverses through link $l \in \mathbf{L}$ is not greater than c_l, the capacity of link l. The *time constraint* means that at any instant of time, there can be only one instance of communication at a given node. To illustrate a fluid model for this constraint, we consider an example shown in Fig. 2.

The network consists of three sessions, S_1, S_2, and S_3, as shown in Fig. 2. Let $x_i,\ i = 1,2,3$, be the data rate of the sessions, respectively. We observe that the time constraint is imposed on each *node* in the network. Let us consider node C in the figure, and define y_{ij}, $i \in \{1,2,3\}$, $j \in \{1,2,3,4\}$, by

$$y_{ij} = \begin{cases} (x_i/c_j), & \text{if } S_i \text{ traverses the link } j \\ 0, & \text{otherwise.} \end{cases}$$

Observe that y_{11} can be interpreted as the *fraction of time* node C expends to receive data of session 1 from node A over a *unit interval of time*. Similarly, y_{13} is interpreted as the fraction of time expended by node C to transmit data of session 1 to node D over a *unit interval of time*. Similar interpretations hold for all y_{ij}. Thus, as total fraction of time expended at node C cannot exceed 1, the *time constraint* at node C is

$$\sum_{i\in\{1,2,3\},j\in\{1,2,3,4\}} y_{ij} \le 1$$

Similar time constraints apply for all other nodes in the network.

TABLE I
LINK AND TIME CONSTRAINTS FOR THE EXAMPLE NETWORK IN FIG. 2

Link	Link Constraint	Node	Time Constraint
1	$\frac{x_1+x_2}{c_1} \le 1$	1	$\frac{x_1+x_2}{c_1} \le 1-\epsilon_1$
2	$\frac{x_2+x_3}{c_2} \le 1$	2	$\frac{x_2+x_3}{c_2} \le 1-\epsilon_2$
3	$\frac{x_1+x_3}{c_3} \le 1$	3	$\frac{x_1+x_2}{c_1}+\frac{x_2+x_3}{c_2}+\frac{x_1+x_3}{c_3} \le 1-\epsilon_3$
4	$\frac{x_1}{c_4} \le 1$	4	$\frac{x_1+x_3}{c_3}+\frac{x_1}{c_4} \le 1-\epsilon_4$
		5	$\frac{x_1}{c_4} \le 1-\epsilon_5$

In general, however, the time constraints presented above are *not sufficient* to ensure that a feasible MAC protocol exists [21], [26]. A feasible MAC always exists if the time constraints are relaxed by replacing the right-hand side of the expressions (i.e., the term "1") by a parameter $\rho \le 2/3$ [26]. In this paper, we consider a fluid model for the MAC, and do not focus on the actual implementation (i.e., MAC protocol) of the resource sharing mechanism at each node. At the fluid time-scale, the details of these different MAC protocols manifest only as an *efficiency* factor that is captured by the parameter ϵ_j, $j \in \mathbf{V}$, which governs the fraction of time that the time resource at each node can be used for successful data transfer. For example, the time constraint at node C is represented by

$$\sum_{i\in\{1,2,3\},j\in\{1,2,3,4\}} y_{ij} \le 1-\epsilon_3.$$

The efficiency factor is chosen such that some MAC protocol is feasible for the given network topology and session configuration. From our earlier discussion, $\epsilon_j \ge 1/3$, $j \in \mathbf{V}$, ensures that a time-division MAC is *always feasible independent of the network topology*. Further, an inefficient MAC scheme would be associated with a larger value of ϵ_j, $j \in \mathbf{V}$. For example, an ideal MAC algorithm would allow the maximum possible (subject to MAC feasibility) time resources at each node to be used for successful data transfer. However, an ALOHA-based MAC would have inefficiencies associated with it, which would allow only a fraction of the time resources at a node to be used for successful data transfer.

Table I presents the link and time constraints for the network in Fig. 2. As we can observe from the table, the link constraints are *subsumed* by the time constraints. Any link constraint is trivially a time constraint, if it is the only flow and terminates at the node. In all other cases, the time constraint is strictly stronger than a link constraint. Thus, we do not need to consider link constraints, and will henceforth restrict ourselves to only time constraints.

C. An Optimization Problem

Let us denote $N(\mathbf{L})$, $N(\mathbf{V})$, and $N(\mathbf{R})$ as the number of links, nodes, and sessions, respectively. For any link l and ses-

sion r, let $\boldsymbol{A}_{lr} = \frac{1}{c_l}$ if link l is in the path of flow r, and 0 otherwise. Thus, we define the matrix $\boldsymbol{A} \in \mathcal{R}^{N(\mathbf{L})\times N(\mathbf{R})}$ by

$$\boldsymbol{A} = \begin{cases} \boldsymbol{A}_{lr} = 1/c_l, & \text{if link } l \text{ in session } r \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

Similarly, define $\boldsymbol{G}_{vl} = 1$ if link l is incident on node v, and 0 otherwise. Thus, the matrix $\boldsymbol{G} \in \{0,1\}^{N(\mathbf{V})\times N(\mathbf{L})}$ is defined by

$$\boldsymbol{G} = \begin{cases} \boldsymbol{G}_{vl} = 1, & \text{if link } l \text{ incident on node } v \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

Using $\boldsymbol{G}$ and $\boldsymbol{A}$, the time constraint for a given network can be expressed as

$$\boldsymbol{GAx} \leq \mathbf{1} - \boldsymbol{\epsilon} \tag{3}$$

where $\boldsymbol{\epsilon} = (\epsilon_j \in [0,1])$, $j \in \mathbf{V}$, and $\boldsymbol{x}$ corresponds to the vector of user data rates. For each user (session) r, let x_r be the data transmission rate. Associated with each user (session) is a utility function $U_r(\cdot)$, which is the "reward" or utility that user r gets by transmitting at the rate of x_r (see [2] for further discussion). Assume that the utility $U_r(x_r)$ is an increasing, strictly concave, and continuously differentiable function of x_r over the range $x_r \geq 0$. In this paper, we restrict ourselves to weighted proportionally fair utility functions of the form $U_r(\cdot) = w_r \log(\cdot)$. From a resource allocation point of view, the resource allocation achieved under any concave and increasing utility functions can be achieved by a weighted proportionally fair allocation[2] [27] through appropriate choice of weights $\{w_r\}$.

The objective is to maximize total utility in the network subject to the link and time constraints. In this paper, we develop congestion control mechanisms to share the time resources in the network in a (weighted) proportionally fair manner. Thus, with session rates $\boldsymbol{x} = (x_r, r \in R)$, we need to solve

$$\mathbf{P} : \max \sum_{r\in R} w_r \log(x_r)$$

subject to

$$\boldsymbol{GAx} \leq \mathbf{1} - \boldsymbol{\epsilon}, \quad \boldsymbol{x} \geq 0.$$

As the cost function is strictly concave and the constraint set is convex, there is a unique solution to $\mathbf{P}$. In Section III, we develop a decentralized congestion control algorithms (both hop-by-hop and end-to-end) to address $\mathbf{P}$.

III. Distributed End-to-End Algorithm

A. Algorithm Description

In this section, we develop an distributed end-to-end congestion control algorithm to solve $\mathbf{P}$. In this paper, we do not consider the optimal global MAC scheduling problem. Thus, "distributed" in this paper means that the congestion controller itself is distributed, since each source needs only local information on resource usage and feedback from the network to adjust its transmission rate. Similar discussion holds for the hop-by-hop congestion control presented in Section IV.

[2]However, the transient dynamics of a decentralized controller may be different.

As the optimization problem has a strictly concave cost function, and convex constraints, we solve $\mathbf{P}$ using Lagrange multipliers. The Lagrangian of the problem $\mathbf{P}$ is

$$L(\boldsymbol{x}, \boldsymbol{\lambda}) = \sum_{r\in R} w_r \log(x_r) - \boldsymbol{\lambda}^T \left(\boldsymbol{GAx} - (\mathbf{1} - \boldsymbol{\epsilon})\right). \tag{4}$$

We denote the input and output link of a session r on the node v when a session r goes through v as $l_i(v,r)$ and $l_o(v,r)$, respectively (for instance, in Fig. 2, $l_i(3,1) = 1$ and $l_o(3,1) = 3$). For completeness, for the source and destination nodes, we define $c_{l_i(s(r),r)} = \infty$ and $c_{l_o(d(r),r)} = \infty$, respectively, where the source and the destination node of session r are denoted by $s(r)$ and $d(r)$. Let us denote $A_j(r)$ as the set of all downstream nodes from node j in the path of session r. Thus, $A_{s(r)}(r)$ is the collection of all nodes in the path of session r.

By differentiating (4), we have

$$\frac{dL}{dx_r} = \frac{w_r}{x_r} - \sum_{j\in A_{s(r)}(r)} \left(\left(\frac{1}{c_{l_i(j,r)}} + \frac{1}{c_{l_o(j,r)}}\right)\lambda_j\right) = 0. \tag{5}$$

Therefore, the unique solution to the problem $\mathbf{P}$ is given by the following condition:

$$x_r = \frac{w_r}{\sum_{j\in A_{s(r)}(r)} \left(\left(\frac{1}{c_{l_i(j,r)}} + \frac{1}{c_{l_o(j,r)}}\right)\lambda_j\right)}. \tag{6}$$

We now present rate adaptation mechanisms for session sources. At each time t, we denote the transmission rate of session r by $x_r(t)$. Suppose that $x_r(t)$ adapts according to

$$\dot{x_r}(t) = \kappa\left(w_r - x_r(t) \sum_{j\in A_{s(r)}(r)} \left(\left(\frac{1}{c_{l_i(j,r)}} + \frac{1}{c_{l_o(j,r)}}\right)\lambda_j(t)\right)\right) \tag{7}$$

where

$$\lambda_j(t) = \beta p_j\left(\sum_{s\in D(j)} x_s(t)\left(\frac{1}{c_{l_i(j,s)}} + \frac{1}{c_{l_o(j,s)}}\right)\right), \tag{8}$$

$p_j(y)$ is a *marking function* at node j, and determines the fraction of flow to be marked. Here, $D(j)$ corresponds to the set of sessions incident on node j, and $\beta > 0$ is a fixed constant.

This function is an indicator of (time) congestion at a node, and sources adapt based on the congestion indication [2], [3]. As in the Internet context, this function is assumed to be a continuous, increasing function with range $[0,1]$.

Observe that (7) is analogous to the differential equation developed in [2]. However, (7) differs from the algorithm of [2] in that (7) handles relative transmission or reception times instead of actual rates. In practice, the incoming/outgoing rates could be known to the nodes by measuring the amount of data over a small interval of time.

To understand the intuition for (7), observe that λ_j is interpreted as the price for using node j per unit time. In addition, $x_r(t)(\frac{1}{c_{l_i(j,r)}} + \frac{1}{c_{l_o(j,r)}})$ is the fraction of time the MAC at node j expends in receiving and re-transmitting (to the next hop) the data from session r. As *time* is the resource in our formulation, the total cost of using node j equals $x_r(t)(\frac{1}{c_{l_i(j,r)}} + \frac{1}{c_{l_o(j,r)}})\lambda_j$.

Thus, the source congestion control mechanism tries to equalize the aggregate cost $x_r(t)\sum_{j\in A_{s(r)}(r)}((\frac{1}{c_{l_i(j,r)}}+\frac{1}{c_{l_o(j,r)}})\lambda_j(t))$ with w_r.

B. Marking Function

Corresponding to each node j in the network is a marking function $p_j(\cdot)$. In this paper, we consider a marking function of the form (at node j)

$$p_j(y)=\left(\frac{y-\tilde{t}_j}{y}\right)^+. \tag{9}$$

This marking function has the interpretation of the fraction of time lost when the time usage at node j exceeds a certain level $\tilde{t}_j$, called the "virtual time" (This is similar to the concept of "virtual capacity" in [3]). As seen in (8), the parameter y of $p_j(y)$ is the sum of MAC time utilizations by *all flows,* both incoming and outgoing, at node j.

Thus, as discussed earlier, $\beta(\frac{1}{c_{l_i(j,r)}}+\frac{1}{c_{l_o(j,r)}})p_j(y)$ marks the fraction of flow which exceeds a *time threshold* $\tilde{t}_j$. Observe that the total time utilization at the MAC cannot exceed 1. Thus, $\tilde{t}_j<1$ is a parameter that *controls the desired time utilization* at the node. For instance, for an inefficient MAC (say, random access), one would set $\tilde{t}_j \ll 1$. If the MAC is more inefficient at node j, the (equilibrium) utilization at that node is smaller. This means that the node has to mark more packets. Thus, an inefficient MAC has low value of $\tilde{t}_j$.

We will discuss the choice of this parameter in Section III-C.

C. Stability Analysis

In this section, we show that the system of controllers defined in (7) is globally stable. The proof is analogous to that in [2]. Let the function $W(\boldsymbol{x})$ be defined by

$$W(\boldsymbol{x})=\sum_{r\in\mathbf{R}} w_r\log x_r - \beta\sum_{j\in\mathbf{V}}\int_0^{\sum_{s\in D(j)} x_s(t)(\frac{1}{c_{l_i(j,s)}}+\frac{1}{c_{l_o(j,s)}})} p_j(y)dy. \tag{10}$$

We can show that $W(\cdot)$ is a strictly concave function, with a unique equilibrium $\boldsymbol{x}^*$.

Then, with a slight extension to [28], we can find virtual times $\{\tilde{t}_j\}$ at each node $j\in\mathbf{V}$, and β, such that the unique maximum of the optimization problem given by (10) also solves $\mathbf{P}$. Thus, the equilibrium rate $\boldsymbol{x}^*$ can be suitably chosen by choosing appropriate values for the time thresholds $\{\tilde{t}_j\}$, and the constant β. In particular, $\{\tilde{t}_j\}$ is chosen such that the equilibrium $\boldsymbol{x}^*$ solves the optimization problem $\mathbf{P}$ discussed in Section II-C. Due to space limitation, we skip the proof. Adaptively choosing these parameters in a manner similar to that in [28], [29] is a topic for future research. We now show that the congestion controllers described by (7) and (8) converge to this equilibrium point. Now, we have the following proposition on the stability of the end-to-end congestion controller.

Proposition 3.1: $w(\boldsymbol{x})$ is a strictly concave, Lyapunov function for the system of differential (7). The unique value of $\boldsymbol{x}$ maximizing $W(\boldsymbol{x})$, denoted by $\boldsymbol{x}^*$ is a stable point of the system, to which all trajectories converge.

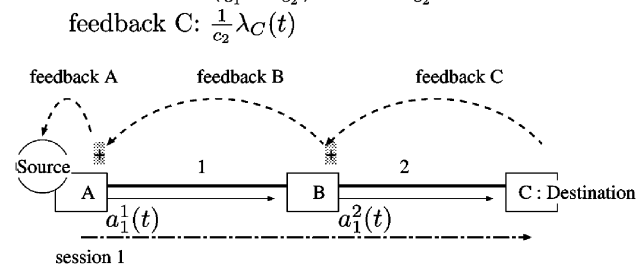


Fig. 3. Hop-by-hop congestion control algorithm.

Proof: The proof is analogous to that in [2], where we use the Lyapunov function defined in (10). We skip the details. ■

IV. Distributed Hop-by-Hop Algorithm

In this section, we develop a distributed hop-by-hop algorithm for congestion control. First, we observe that the congestion controller at the source of each session reacts based on the sum of the congestion prices at each node. Instead of passing this feedback downstream as in the end-to-end algorithm, one could envisage a scheme where each node passes the (partial sum) price upstream. In other words, each node adds its current congestion cost to that it received from a downstream node, and passes this information toward the upstream node. The source will ultimately receive the sum of all price information from the corresponding downstream nodes and use the information for controlling rates. We refer to Fig. 3 for the illustration of the hop-by-hop algorithm.

The basic idea of a hop-by-hop algorithm is that every node in the path of the session operates a congestion control algorithm. In Fig. 3, the congestion price at node C is passed to the upstream node B. Node B computes its local congestion price and adds it to the congestion price from node C. Node B adapts its transmission rate to node C based on this sum of congestion prices. In addition, node B passes this sum of two prices to the upstream node A. Using this "price passing" method, the source of session 1 receives aggregate congestion price from its downstream nodes and controls its transmission rate based on it.

Let us denote $a_r^i(t)$ as the *actual transmission rate* at the ith hop of session r in the hop-by-hop control algorithm. Corresponding to each node i along the path of session r is a *virtual transmission rate* $c_r^i(t)$, which is described by

$$\dot{c}_r^i(t)=\kappa\left(w_r-a_r^i(t)\sum_{j\in A_k(r)}\left(\frac{1}{c_{l_i(j,r)}}+\frac{1}{c_{l_o(j,r)}}\right)\lambda_j(t)\right) \tag{11}$$

$$a_r^i(t)=\min\left[c_r^i(t),a_r^{i-1}(t)\right] \tag{12}$$

where k is the node corresponding to the ith hop of session r. $\{\lambda_j(t)\}$ are defined similar to that in (8), but with the actual transmission rates instead of the source transmission rates. Along the path of each flow r, and for each hop i, the initial conditions for the virtual transmission rates are assumed to satisfy $c_r^i(0)\geq c_r^{i-1}(0)$ (in particular, all of them could be equal).

Thus, in the above algorithm, we sum over all prices downstream along session r. Thus, each node operates a (per-flow) controller based on the perceived congestion due to *downstream nodes*, and determines the maximum rate it can transmit at (the virtual transmission rate). The actual rate it chooses transmits at the rate of the minimum of the *incoming* data rate[3] from $i-1$th hop node in the session's path (the previous hop node), i.e, $a_r^{i-1}(t)$, and the maximum possible rate $c_r^i(t)$.

We comment that at each intermediate node, the controller has knowledge of the local link rates, as well as the "ramp-up" constant w_r for each of the sessions that is incident on the node. It can be shown that the stability analysis and later analysis are valid even if the node uses an upper bound on the ramp-up constant. Thus, from an implementation perspective, one could assume that $\{w_r\}$ are globally bounded by some value w, and use this value at each intermediate node. Heuristically, the convergence proofs are valid even when a bound is used because the data transmission rate into the network is ultimately governed by the source, which will use the correct value of w_r. However, to keep the exposition simple, we will use the exact value of w_r at each node in this paper.

Proposition 4.1: The transmission rates for the hop-by-hop controller described in (11) and (12) converge to the equilibrium value $\boldsymbol{x}^* = (x_1^*, \ldots, x_{N(\mathbf{R})}^*)^T$ given in Proposition 3.1. In particular, for each route r, and for each hop i, $a_r^i(t) \to x_r^*$ as $t \to \infty$.

Proof: At any fixed time t, note that $a_r^i(t)$ decreases along a flow path. This follows from (12), where the $\min$ implies that $a_r^i(t) \le a_r^{i-1}(t)$. Further, from (11), it follows that the sum of the Lagrange multipliers *decreases* along a flow path (as smaller number of nonnegative terms added as we get closer to the destination). These two facts imply that

$$\frac{d}{dt}\left(c_r^i(t) - c_r^{i-1}(t)\right) \ge 0$$

from (11). This observation, along with the ordering of the initial conditions of $\{c_r^i(0)\}$ implies that $c_r^i(t)$ increases along a flow path (i.e., for increasing values of i). In other words, for each i, r, t, we have

$$c_r^i(t) \ge c_r^{i-1}(t). \tag{13}$$

Next, for each i, r, from (12), we have that

$$a_r^i(t) \le c_r^i(t). \tag{14}$$

Thus, from (13) and (14), we have $c_r^i(t) \ge c_r^{i-1}(t) \ge a_r^{i-1}(t)$. Hence, it follows that

$$a_r^i(t) = \min\left\{c_r^i(t), a_r^{i-1}(t)\right\} = a_r^{i-1}(t).$$

This implies that, for each flow r, the actual transmission rates $\{a_r^i(t), i = 1, 2, \ldots\}$ are the same over all hops. This in turn implies that the source dynamics for flow r is the same as source dynamics of the end-to-end controller. Hence, from Proposition 3.1, the result follows. ■

[3]For the source node for each flow, (12) is not considered, as there is no upstream node. Instead we let the actual and virtual transmission rates to be the same.

We finally remark that in practice, $c_r^i(t)$ will be bounded by a finite constant (typically, the output link capacity). This is to ensure that in the absence of congestion, the virtual capacity does not become unbounded (this can happen for instance, if the congestion occurs only upstream to a particular node, and thus, all downstream controllers are redundant).

V. Congestion Control With Delay

In the previous section where we proved stability, we assumed that the time resource was large enough so that queueing did not occur (or equivalently, the time threshold $\tilde{t}$ are suitably chosen). In this section, we do not make such an assumption. We will study the dynamics with queueing in the presence of feedback delay.

For the end-to-end algorithm, we denote the output transmission rate of session r traversing the link l by $x_{r,l}(t)$. Thus, $x_{r,l_i(j,r)}(t)$ and $x_{r,l_o(j,r)}(t)$ are the incoming input and outgoing transmission rate of session r at node j in the end-to-end algorithm, respectively. Similarly, for the hop-by-hop algorithm, we denote the actual and maximum (virtual) sending rate of session r traversing the link l by $a_{r,l}(t)$ and $c_{r,l}(t)$, respectively. Thus, $a_{r,l_i(j,r)}(t)$ and $a_{r,l_o(j,r)}(t)$ are the actual incoming input and actual outgoing transmission rates of session r at node j, respectively.

Finally, each node has a per-flow buffer to temporarily store data before forwarding. We denote the queue length of session r at node j by $q_{rj}(t)$.

A. The End-to-End Controller With Delay

Unlike in the delay-free case considered in Section III, queueing can occur at intermediate nodes due to feedback delay. In this section, we describe the detailed dynamics of rates for a session at each node.

For each node j, let us define $E_I^j(t)$ by

$$E_I^j(t) = \sum_{r \in D(j)} \frac{x_{r,l_i(j,r)}(t)}{c_{l_i(j,r)}}.$$

Thus, $E_I^j(t)$ is the fraction of the time resource at the MAC of node j consumed by incoming flows, and $D(j)$ corresponds to the set of sessions incident on node j. We will assume that the MAC protocol at the nodes ensure that $E_I^j(t) < 1$. Thus, if a timing overload occurs at a node, data loss will occur, causing unsuccessful transmissions to be queued at the preceding hop (where the data was transmitted from). We assume a suitable error and collision detection mechanism exists such that data is queued in case of timing overload. Thus, from a fluid model perspective, we can assume that the successful data transmission into a node j satisfies $E_I^j(t) < 1$. In addition, a poor MAC protocol may not be able to support a time utilization of "1" (for instance an ALOHA-based MAC would have a maximum time utilization less than 0.36). However, in the following discussions, we will assume that the MAC can support a time utilization of "1" for notational ease. The results that are presented can be easily generalized to nonideal MACs by suitable scaling. Let us now define

$$E_O^j(t) = \sum_{r \in D(j)} \frac{x_{r,l_i(j,r)}(t)}{c_{l_o(j,r)}}.$$

The interpretation of $E_O^j(t)$ is the following: If there is no congestion at the node j, the output transmission rates would simply be equal to the incoming rate. $E_O^j(t)$ is the time utilization at the MAC in such a case.

We now consider the following two cases.

1) $E_I^j(t) + E_O^j(t) > 1$
As the time utilization at the node will exceed "1" if the output flow rates equal the input flow rates, we decrease the transmitted output rates such that the time constraint is met. In other words, we choose $\alpha(t) \in (0, 1]$ such that $E_I^j(t) + \alpha(t)E_O^j(t) = 1$, and set the output transmission rate by $x_{r,l_o(j,r)}(t) = \alpha(t)x_{r,l_i(j,r)}(t)$. The remaining flow (of fraction $1 - \alpha(t)$) is queued at node j.

2) $E_I(t) + E_O(t) \leq 1$
In this case, the output flow rate for each session can be set to *at least* the input rate of the corresponding session. If some of the sessions have strictly positive queue lengths, i.e., users with backlogged queues (corresponding to congestion in the past), these users are allocated output rates that are greater than their input rates. The rates will be allocated in some fair manner (for example, a proportional rate increase to all backlogged users), subject to the timing constrain being met. Let us denote $Q_j^+(t)$ be the set of backlogged sessions at node j at time t. We choose $\alpha(t) > 1$ such that the time utilization at the node is less than or equal to one, and for all sessions $r \in Q_j^+(t), x_{r,l_o(j,r)}(t) = \alpha(t)x_{r,l_i(j,r)}(t)$.

B. The Hop-by-Hop Controller With Delay

We now develop the dynamics of the hop-by-hop controller with delay. As in the Section V-A, we define the total time utilization due to incoming flows at node j, by

$$H_I^j(t) = \sum_{r \in D(j)} \frac{a_{r,l_i(j,r)}(t)}{c_{l_i(j,r)}}.$$

Let us denote $Q_j^+(t)$ be the set of backlogged sessions at node j at time t, and define

$$H_O^j(t) = \sum_{\substack{r \in D(j) \\ r \in Q_j^+(t)}} \frac{c_{r,l_o(j,r)}(t)}{c_{l_o(j,r)}} + \sum_{\substack{r \in D(j) \\ r \notin Q_j^+(t)}} \frac{\min[c_{r,l_o(j,r)}(t), a_{r,l_i(j,r)}(t)]}{c_{l_o(j,r)}}$$

where $c_{r,l_o(j,r)}(t)$ is the maximum possible rate for flow r at node k, and is described by (11). We now consider two cases:

1) $H_I^j(t) + H_O^j(t) \leq 1$
In this case, there is no scarce time resource at this node. If the user queues are zero, the output rate is simply equal to the input rate. In general, the output rate for session r is given by

$$a_{r,l_o(j,r)}(t) = \begin{cases} \min[c_{r,l_o(j,r)}(t), a_{r,l_i(j,r)}(t)], & \text{if } q_{rj}(t) = 0 \\ c_{r,l_o(j,r)}(t), & \text{if } q_{rj}(t) > 0. \end{cases}$$

2) $H_I^j(t) + H_O^j(t) > 1$
In this case, the time resource at node j is potentially not sufficient to handle the output rate. Similar to Case 1) for the end-to-end controller in Section V-A, we choose $\alpha(t) \in [0, 1)$ such that $H_I^j(t) + \alpha(t)H_O^j(t) = 1$, and set the output transmission rate correspondingly.

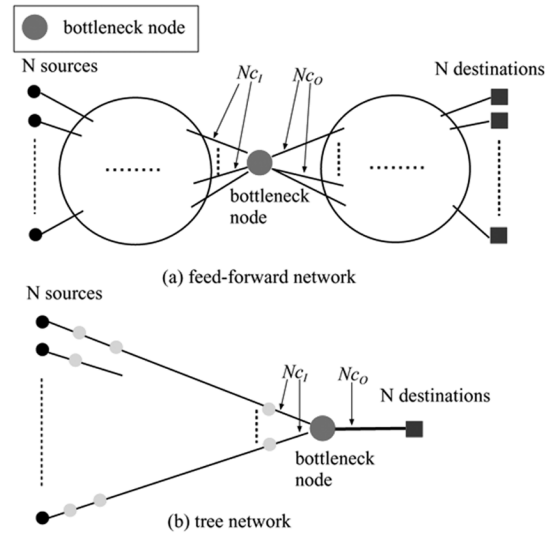


Fig. 4. Networks with spatial spreading: one hop rtt: d_H, and end-to-end rtt of each source: d_R.

VI. Spatial Spreading

In this section, we derive the peak occupied buffer size with the end-to-end controller as well as with the hop-by-hop controller described in Section V. We consider the evolution of these algorithms in the presence of propagation delay. We analytically show the effect of spatial spreading by explicitly deriving the reduction in peak buffer overload under the hop-by-hop scheme.

Consider a feed-forward network, shown in Fig. 4(a), with N sessions where the links are symmetric and have the same propagation delay. Let us denote the link (one-hop) round-trip delay by d_H (i.e., a link propagation delay = $d_H/2$). Further, we assume that all the sessions has the same end-to-end round-trip delay (denoted by d_R). Thus, all sessions have $L = d_R/d_H$ number of hops in their end-to-end path. We assume a single bottleneck in the network (see Fig. 4), and that the delays from the N sources to the bottleneck are all equivalent. We also assume that the capacities of other links than those of input or output links of the bottleneck node are well provisioned, such that congestion occurs only at the common point for all the flows [the bottleneck node in Fig. 4(a)]. We will first consider the end-to-end scheme and compute maximum queue length at the bottleneck node.

Since we consider a system with N flows, we scale the capacities of the bottleneck node with the input and output capacities of the bottleneck node being Nc_I and Nc_O, respectively. This scaling ensures that the *steady-state rate allocated to each user is invariant with the number of sessions.* Physically, this would correspond to a bandwidth scaling at the bottleneck. To mathematically reflect such a scaling, we scale the congestion price appropriately such that the equilibrium rate for each user is invariant with N. In other words, the fraction of flow marked at the bottleneck, $(\frac{1}{Nc_I} + \frac{1}{Nc_O})\lambda^{(N)}(\cdot)$, is invariant with N, where

$\lambda^{(N)}(z) = \beta^{(N)}p^{(N)}(z)$ is the congestion price at the bottleneck in the Nth system; see (8) (i.e., either the marking function or the value of β can be scaled for the Nth system; this is analogous to scaling the marking function in [9], [30]). Hence, the dynamics of each flow $x_j(t)$ at the Nth system are given by

$$\dot{x_j}(t) = \kappa\left[w_j - \beta x_j(t-d_R)\left(\frac{1}{Nc_I}+\frac{1}{Nc_O}\right) \cdot Np\left(\sum_{k=1}^{N} x_k(t-d_R)\left(\frac{1}{Nc_I}+\frac{1}{Nc_O}\right)\right)\right]. \quad (15)$$

Then, the steady-state rate of flow j, denoted by x_j^* is given by

$$x_j^* = \frac{w_j x^*}{\beta(x^*(1/c_I + 1/c_O) - \tilde{t})}$$

where x^* is the average steady-state rate over all flows, and is invariant with N. Note that $\tilde{t}$ is the virtual time constant of the marking function (9), and $p(\cdot)$ is the marking function of the bottleneck node.

Further, we assume that the equilibrium rates are stable, (i.e., $\sum_j x_j^\star(\frac{1}{Nc_I}+\frac{1}{Nc_O}) < 1-\epsilon_b$, where ϵ_b is the MAC efficiency at the bottleneck node). This ensures that the queue lengths are zero at the bottleneck link when the system is at equilibrium (however, queues will build up during transients due to network propagation delay).

We finally comment that we have assumed that the feedback (marks) do not experience congestion, and that the delay in the feedback is solely due to propagation delays. As we have per-flow queueing, a packet implementation to approximate this could be the following. When congestion occurs at a node, instead of marking the incoming packet (implemented via setting the ECN (Explicit Congestion Notification) bit [31]), one could mark the head-of-line (outgoing) packet in the queue of the corresponding user. This would ensure that the queueing delays are minimized for the feedback, and that the source gets the appropriate feedback. Such a scheme seems feasible in a wireless context, as per-flow queueing is expected to be implemented for scheduling as well as physical layer reasons [12], [18], [19]. Further, the number of flows in this network is expected to be of a smaller scale than that in the wired network.

Let $x(t) = \frac{1}{N}\sum_{j=1}^{N} x_j(t)$ and $w = \frac{1}{N}\sum_{j=1}^{N} w_j$. By summing (15) across all sessions, we obtain

$$\begin{aligned}\dot{x}(t) &= \kappa\left[w - \beta\left(x(t-d_R)\left(\frac{1}{c_I}+\frac{1}{c_O}\right) - \tilde{t}\right)^+\right] \\ &= \tilde{\kappa}[\tilde{w} - (x(t-d_R) - \tilde{c})^+] \end{aligned} \quad (16)$$

where $\tilde{c} = \frac{\tilde{t}}{(\frac{1}{c_I}+\frac{1}{c_O})}$, $\tilde{\kappa} = \kappa(\frac{1}{c_I}+\frac{1}{c_O}))\beta$, and $\tilde{w} = w/(\beta(\frac{1}{c_I}+\frac{1}{c_O}))$.

Next, for each time t, under the end-to-end control scheme, let us denote the average (over flows) queue length (across sessions) at the bottleneck node by $q^e(t)$, and the average input and output rates by $x_I(t)$ and $x_O(t)$, respectively. Observe that congestion occurs if $\frac{x_I(t)}{c_I^b} + \frac{x_O(t)}{c_O^b} > 1$, where $c_I^b = c_I/(1-\epsilon_b)$ and $c_O^b = c_O/(1-\epsilon_b)$. Since the constant scaling of c_I and c_O does not affect the analysis result (Lemma 6.1 and Proposition 6.1), we assume that $\epsilon_b = 1$ for notational simplicity.

Further, observe that $x_I(t) \le c_I$. We now describe the dynamics of the queue length process. We consider several cases:

1) $\frac{c_I c_O}{c_I+c_O} < x_I(t) \le c_I$

$$\begin{aligned}\dot{q}^e(t) &= x_I(t) - x_O(t) = c_I\frac{x_I(t)}{c_I} - c_O\left(1-\frac{x_I(t)}{c_I}\right) \\ &= \left(\frac{c_I+c_O}{c_I}\right)\left[x_I(t) - \frac{c_I c_O}{c_I+c_O}\right]\end{aligned} \quad (17)$$

2) $x_I(t) \le \frac{c_I c_O}{c_I+c_O}$ and $q^e(t) > 0$
The dynamics of $\dot{q}^e(t)$ are identical to that in Case 1).

3) $x_I(t) \le \frac{c_I c_O}{c_I+c_O}$ and $q^e(t) = 0$
In this case, as the buffer at the bottleneck node is empty, and there is no congestion, we have $\dot{q}^e(t) = 0$.

Thus, with

$$\dot{\tilde{q}}^e(t) = \frac{\dot{q}^e(t)}{(c_I+c_O)/c_I} \quad (18)$$

we have

$$\dot{\tilde{q}}^e(t) = \begin{cases} x_I(t) - c, & \text{if } \tilde{q}^e(t) > 0 \\ (x_I(t) - c)^+, & \text{if } \tilde{q}^e(t) = 0. \end{cases} \quad (19)$$

where $c = \frac{c_I c_O}{c_I+c_O}$. As discussed earlier, we assume that the equilibrium rate is stable. Thus, we let γ be the node utilization at the bottleneck node, i.e., $x^\star = \gamma c$, where $0 < \gamma < 1$.

We next derive the "worst case" peak initial queue length at the bottleneck node under the end-to-end controller as well as a hop-by-hop controller, and with the system starting from rest (i.e., the value of the first local maximum of the queue length at the bottleneck, when all initial conditions are zero). Let us define ${Q_{\max}}^e(H,\eta)$ to be the (unscaled) initial maximum queue length at the bottleneck node in the end-to-end control with the one-hop round-trip delay and the number of hops (links) for each session being η and H, respectively (thus, the end-to-end round-trip delay is ηH). Also let ${q_{\max}}^e(H,\eta) = {Q_{\max}}^e(H,\eta)/N$ be the peak (initial) queue length for the scaled system defined by (16) and (19).

Let

$$\begin{aligned}\bar{q}^e_{\max}(1,d) &= \left(\tilde{\kappa}\tilde{w} + \left(\frac{\tilde{\kappa}\tilde{w}}{c(1-\gamma)}\right)^2\right)d^2 \\ &+ \left(c(1-\gamma) + \frac{\tilde{\kappa}\tilde{w}}{c(1-\gamma)}\right)d + \frac{c^2(1-\gamma)^2}{\tilde{\kappa}\tilde{w}} + c(1-\gamma). \end{aligned} \quad (20)$$

With this definition, we have:

Lemma 6.1: Fix any $d > 0$. Then, we have $q^e_{\max}(1,d) \le \bar{q}^e_{\max}(1,d)$, and for any $\alpha \in (0,1)$, $\exists M_o$ with $M_o \ge 1$, such that $\forall\, M \ge M_o$, $M^\alpha \bar{q}^e_{\max}(1,d) \le {q_{\max}}^e(M,d)$.

The proof consists of two parts. We first derive an upper bound on the queue length for a one-hop network (with delay d) by 1) observing that the source controller increase rate cannot increase faster than w (i.e., $\dot{x}(t)) \le w$), and 2) by providing a lower bound on the rate of decrease of the arrival rate, once the controller rate has peaked. More precisely, for time $t \ge t_5 + d$, where t_5 is the time at which the controller rate peaks (see Fig. 6), we show $\dot{x}(t) \le -\epsilon$ for some explicitly constructed $\epsilon > 0$. This upper bound is denoted by $\bar{q}^e_{\max}(1,d)$, and formally given by (20). We next derive a lower bound on ${q_{\max}}^e(M,d)$, by appropriately bounding the end-controller transmission rates.

Using this lower bound, and the upper bound for the one-hop system, the lemma is proved. The details of the proof are presented in the Appendix.

Using this result, we prove the main result of this section. Similar to $Q_{\max}{}^e(H,\eta)$ and $q_{\max}{}^e(H,\eta)$, let us define $Q_{\max}{}^h(H,\eta)$ to be the unscaled maximum queue length (due to initial transients) with the *hop-by-hop control*, and $q_{\max}{}^h(H,\eta)$ to be the corresponding scaled queue length. We then have

Proposition 6.1: Fix any $d > 0$. Then, for any $\alpha \in (0,1)$, $\exists M_o$ with $M_o \geq 1$, such that $\forall\ M \geq M_o$, $q^h_{\max}(M,d) \leq q^e_{\max}(M,d)/M^\alpha$.

Proof: The proof proceeds analogously to that in Lemma 6.1. As the control loop for the hop-by-hop controller has round-trip delay d, the upper bound derived for the one-hop end-to-end controller with round-trip delay of d can be shown to also hold here, i.e., $q_{\max}{}^h(M,d) \leq \bar{q}^e_{\max}(1,d)$. Now, from Lemma 6.1, the desired result follows. ■

Remark 6.1: The tree network in Fig. 4(b) is a special case of the feed-forward network, such that each session has a separate path. The tree architecture could be used in a community roof-top wireless network, where a single wired based station could be used to provide community Internet access. Tree architectures are also popular in Bluetooth-based ad hoc networks [32], [33], as well as in broadcasting applications [34]. In Section VII, we will see that we achieve significant gains for the tree network even with only five flows.

VII. Simulation Results

In this section, we present simulation results that compare the hop-by-hop algorithm with the end-to-end algorithm, and show that there is a significant decrease in the peak load with the hop-by-hop algorithm. We consider packet level simulations using the ns-2 [35] simulator. We have made suitable modifications in ns-2 to support hop-by-hop as well as end-to-end controllers with a MAC.

A. Simulation Setup

The network topology used in the simulation is a tree network shown in Fig. 4(b), with five sessions and five hops from the sources and the destinations. Each of the input and output links at the bottleneck node has a bandwidth of 5 Mb/s. Since we use a fixed size packet of 1000 bytes, this corresponds to 625 packets per second. Thus, the equilibrium rate over one session at the bottleneck is 62.5 packets per second.

In our ns-2 implementation, we have assumed a separate side channel for communicating prices (with a bit rate of 1 bit per data packet) that enables a simple implementation. In practice, this price information can be embedded in ACK packets. We used a fixed data packet size of 1000 bytes. For every one data packet, the associated ACK packet contains one ECN bit, which represents marked/unmarked status. To compute the price information at every node, each node measures outgoing/incoming data rate over all incident links on the corresponding node in the following manner. For outgoing rates, the node knows the transmission rates (from the congestion controller at the node), and for incoming rates, the node measures rates by computing inter-packet times (using a sliding window mechanism).

The following parameters have been chosen for the congestion controllers and marking functions at the intermediate routers, such that the node utilization ratio is about 95%: $\kappa = 0.5$, $w = 21$, $\tilde{t} = 0.25$, and $\beta = 30$. From (15), we have

$$x^\star\left(\frac{1}{c_I} + \frac{1}{c_O}\right) = \tilde{t} + \frac{w}{\beta} = 0.25 + \frac{21}{30} = 0.95. \qquad (21)$$

Thus, we have the 95% node utilization.

In our implementation, we approximate the fluid model for a MAC with a time-division MAC. The MAC protocol implemented in the simulation is a simple centralized TDMA MAC protocol (a centralized scheduler that distributes the requested number of time slots to each link). A time slot interval is chosen to be the transmission time of a single packet. If the incoming/outgoing traffic imposed on a node violates the timing constraint, as discussed in Section V in the paper, we give higher priority to the incoming traffic to the nodes (in this case queueing occurs on the output link buffer).

B. Simulation Results

Fig. 5(a) and (c) plots the instantaneous sum rate of five sessions for 10 ms and 200 ms of one-hop delay, respectively. The 10-ms case corresponds to an efficient MAC, where the hop-delay primarily occurs due to the propagation delay and physical layer air interface. On the other hand, the case where the hop delay is 200 ms corresponds to a network with large per-hop delays (possibly due to the MAC implementation). In both of these cases, Fig. 5(a) and (c), we see that the transmission rates (with the end-to-end and hop-by-hop controllers) oscillate about the corresponding fixed point. As expected, we observe that for the large delay case, the end-to-end algorithm leads to larger oscillations than the hop-by-hop algorithm.

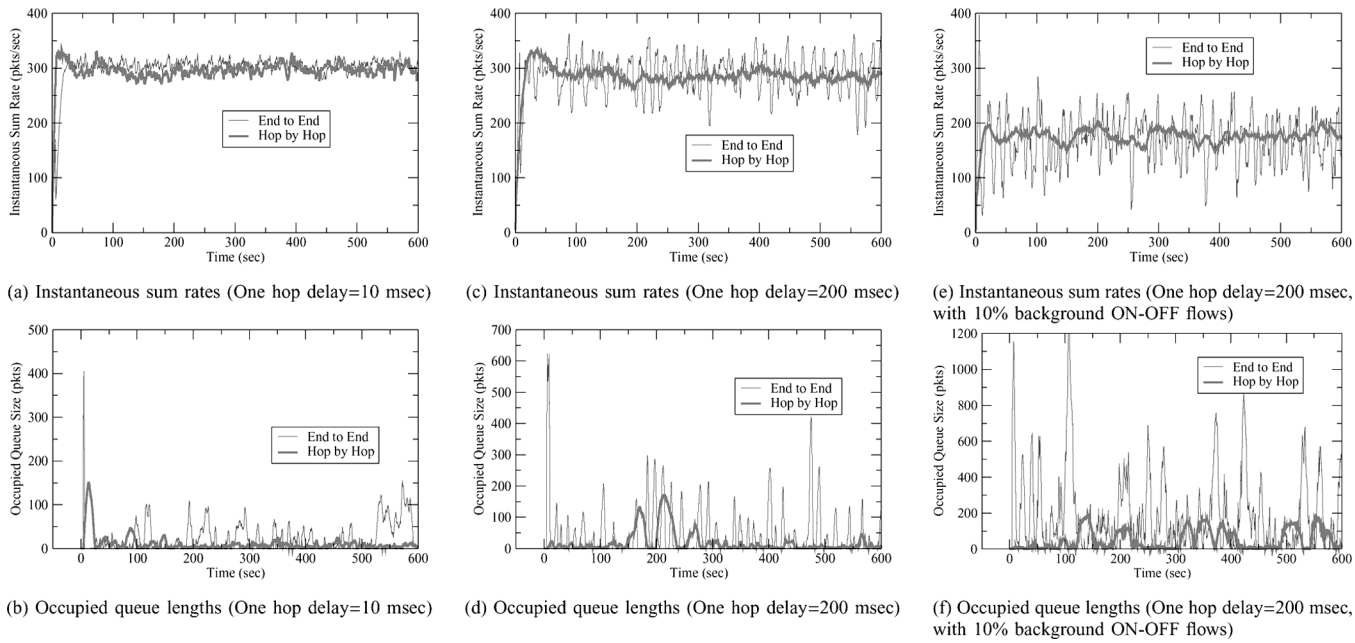
In Fig. 5(b) and (d), we plot the evolution of the bottleneck queue length (measured every 20 ms). The spatial spreading effect is clearly illustrated by these plots, where we see that the peak buffer size with the hop-by-hop controller is much smaller than that of the end-to-end controller. These fluid and packet simulations indicate that significant gains are to be had with the hop-by-hop scheme, and validate our analytical results.

We also provide a plot with short background flows (modeled by ON–OFF processes) in Fig. 5(e) and (f). The only difference from the previous setup is that we have background ON–OFF flows, whose mean rate is set to be 10% of the equilibrium rates of controlled flows, and their burst and idle times are set to be 50 ms each. Given a w, we suitably chosen β and $\tilde{t}$ such that the node utilization is about 95%. The values are chosen in a similar manner to (21). Fig. 5(e) and (f) show that again, the hop-by-hop scheme outperforms an end-to-end scheme.

Appendix
Proof of Lemma 6.1

Proof: In this proof, we will show the following: For any fixed $d > 0$,

1) $q_{\max}{}^e(1,d) \leq \bar{q}^e_{\max}(1,d)$
2) $\exists M_o$ such that $\forall\ M \geq M_o$, $q_{\max}{}^e(M,d) \geq KMd$, where K is a positive finite constant.



(a) Instantaneous sum rates (One hop delay=10 msec)
(c) Instantaneous sum rates (One hop delay=200 msec)
(e) Instantaneous sum rates (One hop delay=200 msec, with 10% background ON-OFF flows)
(b) Occupied queue lengths (One hop delay=10 msec)
(d) Occupied queue lengths (One hop delay=200 msec)
(f) Occupied queue lengths (One hop delay=200 msec, with 10% background ON-OFF flows)

Fig. 5. Instantaneous sum rates and occupied queue lengths (at the bottleneck node) for both end-to-end and hop-by-hop controllers.

Suppose that 1) and 2) are true. Then, from the definition of $\bar{q}^e_{\max}(1,d)$ [see (20)], for any $0<\alpha<1$, and sufficiently large M, we have

$$\begin{aligned} M^\alpha {q_{\max}}^e(1,d) &\le M^\alpha \bar{q}^e_{\max}(1,d) \\ &\le KMd \le {q_{\max}}^e(M,d). \end{aligned}$$

Thus, it suffices to show 1) and 2) to complete the proof.

Proof of 1): Consider a network of a single link with the end-to-end round-trip delay of each session being d. As discussed earlier, we have the equilibrium rate $x^\star = \gamma c$, where γ is the node utilization with $0<\gamma<1$. Thus, at the steady state we have

$$\tilde{w} = \gamma c p(\gamma c) = \gamma c - \tilde{c}. \tag{22}$$

Recall that $p(\cdot)$ is the marking function of the bottleneck node [see (15)].

Let us define the following time epochs (see Fig. 6):

$$\begin{aligned} &t_1 = \inf\{t>0 : x(t) > \tilde{c}\}, \quad t_2 = \inf\{t > t_1 : x(t) > \gamma c\}, \\ &t_3 = \inf\{t > t_2 : x(t) > c\}, \quad t_4 = t_1 + d, \quad t_5 = t_2 + d, \\ &t_6 = t_4 + d, \quad t_7 = \inf\{t > t_5 : x(t) < c\}. \end{aligned} \tag{23}$$

We first consider an end-to-end system with an input rate constraint c_I (which could be arbitrarily large), as shown in Fig. 6(a). Due to the input constraint c_I, the actual rate arriving at the bottleneck node, which we denote by $x_{c_I}(t)$, could be different from $x(t)$, depending on c_I and R_A [see Fig. 6(a)]. We will upper-bound $x(t)$ (and $x_{c_I}(t)$) by a trajectory $\bar{x}(t)$, and use this bound to compute an upper bound on the peak queue length. We consider the following two cases: a) $c_I \ge R_A$ and b) $c_I < R_A$.

Case a): $c_I \ge R_A$

In this case, $x(t)$ is the actual input arrival rate at the bottleneck node. Now, observe that the peak queue length at the bottleneck node is given by

$${q_{\max}}^e(1,d) = \int_{t_3}^{t_7} (x(t)-c)dt.$$

We assume that the initial condition satisfies $x(s) \le \tilde{c}, \forall\, s \le 0$.

We can see that $x(t)$ achieves the maximum (i.e., R_A) at t_5, since $\dot{x}(t_5) = 0$. This follows from the fact that

$$\dot{x}(t_5) = \tilde{\kappa}(\tilde{w} - x(t_2)p(x(t_2))) = \tilde{\kappa}(\tilde{w} - \gamma c p(\gamma c)) = 0$$

and from (22). For $t \in [t_1, t_4]$, let $\bar{x}(t) = x(t)$. Then, by the assumption on the initial condition

$$\dot{x}(t) = \dot{\bar{x}}(t) = \tilde{\kappa}\tilde{w}, \quad t \in [t_1, t_4]. \tag{24}$$

Next, let $\dot{\bar{x}}(t) = \tilde{\kappa}\tilde{w}, t \in (t_4, t_5]$ with $\bar{x}(t_4) = R_B$ (i.e., straight-line extension of $\bar{x}(t)$, $t \in [t_1, t_4]$ until t_5). Then, we have

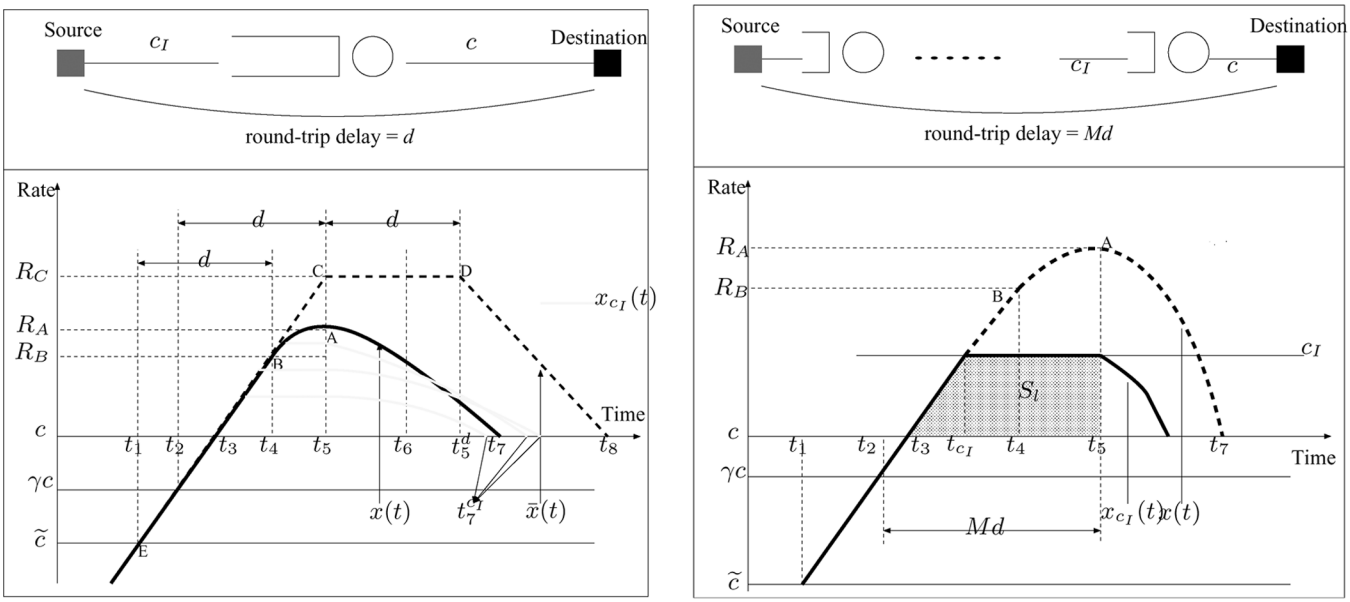
$$x(t) < \bar{x}(t), \quad t \in (t_4, t_5] \tag{25}$$

since for $t \in (t_4, t_5]$

$$\begin{aligned} \dot{x}(t) &= \tilde{\kappa}(\tilde{w} - x(t-d)p(x(t-d)) \\ &< \tilde{\kappa}\tilde{w} = \dot{\bar{x}}(t). \end{aligned}$$

Let $t_5^d = t_5 + d$. Further, for $t \in (t_5, t_5^d]$ let $\bar{x}(t) = \bar{x}(t_5)$. Then, we have

$$x(t) \le R_A < \bar{x}(t_5) = \bar{x}(t), \quad t \in \left(t_5, t_5^d\right]. \tag{26}$$



(a) Upper bound on peak queue length $q_{max}{}^e(1,d)$

(b) Lower bound on peak queue length $q_{max}{}^e(M,d)$

Fig. 6. Upper bound and lower bound on peak queue length.

Note that we have the following two cases: (i) $t_5^d \le t_7$ and (ii) $t_5^d > t_7$. First, consider the case of $t_5^d \le t_7$ [Fig. 6(a)]. For $t \in (t_5^d, t_7]$, let us define $\bar{x}(t)$ with $\dot{\bar{x}}(t) = \tilde{\kappa}(\tilde{w} - (c - \tilde{c}))$.

Then, using the fact that $x(t-d) > c$, $t \in \left(t_5^d, t_7\right]$, we have $\dot{\bar{x}}(t) > \dot{x}(t) = \tilde{\kappa}(\tilde{w} - (x(t-d) - \tilde{c}))$, leading to the fact that

$$x(t) < \bar{x}(t), \quad t \in \left(t_5^d, t_7\right]. \tag{27}$$

Second, if $t_5^d > t_7$, this case corresponds to (26).

Case b) $c_I < R_A$

In this case, $x_{c_I}(t)$ is the actual input arrival rate at the bottleneck node. Let $t_7^{c_I} = \inf\{t > t_5 : x_{c_I}(t) < c\}$. Note, however, that $\forall\, t \in [t_3, t_7^{c_I}]$, we have $x_{c_I}(t) \ge c$. Thus, the bound (i.e., $\bar{x}(t)$) constructed in Case (i) based on $x(t)$ also holds for $x_{c_I}(t)$, i.e.,

$$x_{c_I}(t) \le \bar{x}(t), \quad t \in [t_3, t_7^{c_I}]. \tag{28}$$

From (25), (26), (27), and (28), $\bar{x}(t)$ is an upper bound on $x(t)$ or $x_{c_I}(t)$ irrespective of the position of c_I.

By the straight-line extension of $\bar{x}(t)$ until it crosses c, we define $t_8 \triangleq \inf\{t > t_5^d : \bar{x}(t) < c\}$.

Note that we have

$$\begin{aligned} t_5 - t_3 &= d - (t_3 - t_2) = d - \frac{c(1-\gamma)}{\tilde{\kappa}\tilde{w}} \\ R_C - c &= \tilde{\kappa}\tilde{w}(t_5 - t_3) \\ t_8 - t_5^d &= \frac{R_C - c}{\tilde{\kappa}(c - \tilde{c} - \tilde{w})} = \frac{R_C - c}{c(1-\gamma)}. \end{aligned}$$

Thus, the upper bound on the peak occupied queue length for a fixed d is given by

$$\begin{aligned} q_{\max}^e(1,d) &\le \int_{t_3}^{t_8} (\bar{x}(t) - c)dt \\ &= \frac{1}{2}(t_5 - t_3 + t_5^d - t_5 + t_8 - t_5^d)(R_C - c) \\ &\le \left(\tilde{\kappa}\tilde{w} + \left(\frac{\tilde{\kappa}\tilde{w}}{c(1-\gamma)}\right)^2\right) d^2 \\ &\quad + \left(c(1-\gamma) + \frac{\tilde{\kappa}\tilde{w}}{c(1-\gamma)}\right) d \\ &\quad + \frac{c^2(1-\gamma)^2}{\tilde{\kappa}\tilde{w}} + c(1-\gamma) \\ &= \bar{q}_{\max}^e(1,d). \end{aligned} \tag{29}$$

This completes the proof of (i).

Proof of (ii): Next, we show (ii), which describes $q_{\max}{}^e(M,d)$ for *sufficiently large* M. In this setup, we consider an end-to-end controlled system with its round-trip delay being Md. Thus, the time epochs described in (23) hold for the round-trip delay Md. Fig. 6(b) explains this setup.

Recall that $R_B = x(t_4)$. As $R_B - \tilde{c} = (t_4 - t_1)\tilde{\kappa}\tilde{w}$, we have

$$R_B = Md\tilde{\kappa}\tilde{w} + \tilde{c}. \tag{30}$$

Note that we derive $q_{\max}{}^e(M,d)$ for *sufficiently large* M. Then, it is clear that for a sufficiently large M and any finite c_I, we have $c_I \le R_B$. Thus, $x_{c_I}(t)$ is the actual arrival rate to the bottleneck node with its input constraint of c_I. Instead of computing the exact peak queue length, we lower-bound it by computing the area of the shaded region (denoted by S_l) in Fig. 6(b).

Let t_{c_I} be the first time after t_3 such that $\bar{x}(t)$ hits c_I. By definition of time epochs, $t_5 - t_3 = Md$ [see (23)], and we have

$$t_{c_I} - t_3 = \frac{c_I - c}{\tilde{\kappa}\tilde{w}}.$$

Then, we have

$$\begin{aligned} S_I &= \frac{1}{2}\left(2(t_5 - t_3) - (t_{c_I} - t_3)\right)(c_I - c) \\ &= \frac{1}{2}\left(2Md - \frac{c_I - c}{\tilde{\kappa}\tilde{w}}\right)(c_I - c) \\ &= Md(c_I - c) - \frac{1}{2}\frac{(c_I - c)^2}{\tilde{\kappa}\tilde{w}}. \end{aligned} \tag{31}$$

Thus, we can find a positive constant K, such that $\forall\, M \geq M_o$

$$q^e_{\max}(M, d) \geq KMd. \tag{32}$$

Then, the proof of (ii) follows from (32).

ACKNOWLEDGMENT

The authors would like to thank Prof. G. de Veciana for his comments and valuable discussions.

REFERENCES

[1] G. Holland and N. H. Vaidya, "Analysis of TCP performance over mobile ad hoc networks," *Proc. IEEE/ACM Mobicom*, pp. 219–230, Aug. 1999.

[2] F. P. Kelly, A. Maulloo, and D. Tan, "Rate control in communication networks: Shadow prices, proportional fairness and stability," *J. Oper. Res. Soc.*, vol. 49, pp. 237–252, 1998.

[3] S. Kunniyur and R. Srikant, "End-to-end congestion control: Utility functions, random losses and ECN marks," in *Proc. IEEE INFOCOM*, 2000, vol. 3, pp. 1323–1332.

[4] G. Vinnicombe, On the stability of end-to-end congestion control for the Internet Univ. Cambridge, Cambridge, U.K., Tech. Rep., 2001.

[5] F. P. Kelly, "Models for a self-managed internet," *Philosoph. Trans. Royal Soc.*, vol. A358, pp. 2335–2348, 2000.

[6] L. Massoulie, "Stability of distributed congestion control with heterogenous feedback delays," Microsoft Research, Cambridge, U.K., Tech. Rep., 2000.

[7] F. Paganini, J. Doyle, and S. Low, "Scalable laws for stable network congestion control," in *Proc. IEEE Conf. Decision and Control*, 2001, vol. 1, pp. 185–190.

[8] R. Johari and D. Tan, "End-to-end congestion control for the internet: Delays and stability," *IEEE/ACM Trans. Networking*, vol. 9, no. 6, pp. 818–832, Dec. 2001.

[9] S. Shakkottai, R. Srikant, and S. Meyn, "Bounds on the throughput of congestion controllers in the presence of feedback delay," *IEEE/ACM Trans. Networking*, vol. 11, no. 6, pp. 972–981, Dec. 2003.

[10] S. H. Low and D. E. Lapsley, "Optimization flow control, I: Basic algorithm and convergence," *IEEE/ACM Trans. Networking*, vol. 7, no. 6, pp. 861–875, Dec. 1999.

[11] S. H. Low, "A duality model of TCP and queue management algorithms," *IEEE/ACM Trans. Networking*, vol. 11, no. 4, pp. 525–536, Aug. 2003.

[12] P. Bender, P. Black, M. Grob, R. Padovani, N. Sindhushayana, and A. Viterbi, "CDMA/HDR: A bandwidth efficient high speed wireless data service for nomadic users," *IEEE Commun. Mag.*, vol. 38, no. 7, pp. 70–77, Jul. 2000.

[13] *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Standard 802.11, 1997.

[14] P. P. Mishra and H. Kanakia, "A hop by hop rate based congestion control scheme," in *Proc. ACM SIGCOMM*, Aug. 1992, pp. 112–123.

[15] H. T. Kung, T. Blackwell, and A. Chapman, "Credit-based flow control for ATM networks: Credit update protocol, adaptive credit allocation and statistical multiplexing," in *Proc. ACM SIGCOMM*, 1994, pp. 101–114.

[16] L. Tassiulas, "Adaptive back-pressure congestion control based on local information," *IEEE Trans. Automat. Contr.*, vol. 40, no. 2, pp. 236–250, Feb. 1995.

[17] S. Sarkar and L. Tassiulas, "Back pressure based multicast scheduling for fair bandwidth allocation," in *Proc. IEEE INFOCOM*, 2001, pp. 1123–1132.

[18] S. Borst, "User-level performance of channel-aware scheduling algorithms in wireless data networks," in *Proc. IEEE INFOCOM*, 2003, pp. 321–331.

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

[20] A. Kortebi, L. Muscariello, S. Oueslati, and J. Roberts, "On the scalability of fair queueing," presented at the ACM HotNets-III Conf., San Diego, CA, Nov. 2004.

[21] L. Tassiulas and S. Sarkar, "Maxmin fair scheduling in wireless networks," in *Proc. IEEE INFOCOM*, 2002, pp. 763–772.

[22] X. Lin and N. B. Shroff, "Joint rate control and scheduling in multihop wireless networks," in *Proc. IEEE Conf. Decision and Control*, 2004, pp. 1484–1489.

[23] L. Chen, S. H. Low, and J. C. Doyle, "Joint congestion control and media access control design for ad hoc wireless networks," in *Proc. IEEE INFOCOM*, 2005, pp. 2212–2222.

[24] T. S. Rappaport, *Wireless Communications: Principles and Practice*. Upper Saddle River, NJ: Prentice-Hall, 2002.

[25] M. Kodialam and T. Nandagopal, "Characterizing achievable rates in multi-hop wireless networks: The joint routing and scheduling problem," in *Proc. ACM MobiCom*, 2003, pp. 42–54.

[26] B. Hajek and G. Sasaki, "Link scheduling in polynomial time," *IEEE Trans. Inf. Theory*, vol. 34, no. 5, pp. 910–917, Sep. 1988.

[27] J. Mo and J. Walrand, "Fair end-to-end window-based congestion control," *IEEE/ACM Trans. Networking*, vol. 8, no. 5, pp. 556–567, Oct. 2000.

[28] S. Kunniyur and R. Srikant, "A time-scale decomposition approach to adaptive ECN marking," in *Proc. IEEE INFOCOM*, 2001, pp. 1330–1339.

[29] ——, "Analysis and design of an adaptive virtual queue algorithm for active queue management," in *Proc. ACM SIGCOMM*, Aug. 2001, pp. 123–134.

[30] S. Shakkottai and R. Srikant, "How good are deterministic fluid models of internet congestion control?," in *Proc. IEEE INFOCOM*, 2002, pp. 497–505.

[31] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Networking*, vol. 1, no. 4, pp. 397–413, Aug. 1993.

[32] G. V. Zaruba, S. Basagni, and I. Chlamtac, "Bluetrees-scatternet formation to enable Bluetooth-based ad hoc networks," in *Proc. IEEE Int. Conf. Communications (ICC)*, 2001, pp. 273–277.

[33] R. Guerin, J. Rank, S. Sarkar, and E. Vergetis, "Forming connected topologies in Bluetooth ad hoc networks," in *Proc. 18th Int. Teletraffic Congress (ITC-18)*, 2003, pp. 1011–1020.

[34] J. E. Wieselthier, G. D. Nguyen, and A. Ephremides, "On the construction of energy-efficient broadcast and multicast trees in wireless networks," in *Proc. IEEE INFOCOM*, 2000, pp. 585–594.

[35] Network Simulator ns-2. [Online]. Available: http://www.isi.edu/nsnam/ns/

Yung Yi (S'04–M'07) received the B.S.E. and M.S.E. degrees in computer science and engineering from Seoul National University, Seoul, Korea, in 1997 and 1999, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Texas at Austin in 2006.

He is currently a Post-Doctoral Research Associate in the Department of Electrical Engineering at Princeton University, Princeton, NJ. His current research interests include scheduling and QoS for wireless networks, congestion control in the Internet, and design and performance analysis of computer networks.

Sanjay Shakkottai (M'02) received the Ph.D. degree from the University of Illinois at Urbana-Champaign in 2002.

He is currently an Assistant Professor in the Department of Electrical and Computer Engineering, University of Texas at Austin. His research interests include wireless and sensor networks, stochastic processes and queueing theory.

Dr. Shakkottai was the Finance Chair of the 2002 IEEE Computer Communications Workshop in Santa Fe, NM. He received the National Science Foundation CAREER Award in 2004.