Fundamental Trade-Offs in Aggregate Packet Scheduling

Zhenhai Duan, *Member, IEEE*, Zhi-Li Zhang, *Member, IEEE*, Yiwei Thomas Hou, *Senior Member, IEEE*

Abstract

In this paper we investigate the fundamental trade-offs in aggregate packet scheduling for support of guaranteed delay service. In our study, we consider two classes of aggregate packet scheduling algorithms: the *static earliest time first* (SETF) and *dynamic earliest time first* (DETF). Through these two classes of aggregate packet scheduling (and together with the simple FIFO packet scheduling algorithm), we show that, with additional time stamp information encoded in the packet header for scheduling purpose, we can significantly increase the maximum allowable network utilization level, while at the same time reducing the worst-case edge-to-edge delay bound. Furthermore, we demonstrate how the number of the bits used to encode the time stamp information affects the trade-off between the maximum allowable network utilization level and the worst-case edge-to-edge delay bound. In addition, the more complex DETF algorithms have far superior performance than the simpler SETF algorithms. These results illustrate the fundamental trade-offs in aggregate packet scheduling algorithms and shed light on their provisioning power in support of guaranteed delay service.

Index Terms

Packet Scheduling Algorithms, Aggregate Packet Scheduling Algorithms, Quality of Services (QoS), Performance Analysis

I. Introduction

Because of its potential scalability in support of Internet QoS guarantees, lately aggregate packet scheduling has attracted a lot of attention in the networking community. For instance, in the DiffServ framework [2], it is proposed that the simple FIFO packet scheduling be used to support the EF (expedited forwarding) per-hop behavior (PHB) [8]. Namely, at each router, EF packets from all users are queued at a single FIFO buffer and serviced in the order of their arrival times at the queue. Clearly, use of FIFO packet scheduling results in a very simple implementation of the EF PHB. However, the ability of appropriately provisioning a network using FIFO packet scheduling to provide guaranteed rate/delay service—as the EF PHB is arguably intended to support [9]—has been questioned [1], [6].

An earlier, abridged version of this paper appeared in the Proc. IEEE ICNP'2001 under the same title.

Zhenhai Duan is with the Computer Science Department, Florida State University, Tallahassee, FL 32306 USA (e-mail: duan@cs.fsu.edu).
Zhi-Li Zhang is with the Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN 55455 USA (e-mail: zhzhang@cs.umn.edu).
Yiwei Thomas Hou is with the Bradley Department of Electrical and Computer Engineering, Virginia Tech, Blacksburg, VA 24061 USA (e-mail: thou@vt.edu).

In a recent work by Charny and Le Boudec [6], it is shown that in order to provide guaranteed delay service using FIFO, the overall network utilization level should be limited to a small fraction of its link capacities. More specifically, in a network of FIFO schedulers, a bound on the worst-case delay at each router is derived only when the network utilization level is limited to a factor smaller than $1/(H^* - 1)$, where H^*, referred to as the *network diameter*, is the number of hops in the longest path of the network. Furthermore, given the network utilization level $\alpha < 1/(H^* - 1)$, the derived worst-case delay bound is inversely proportional to $1 - \alpha(H^* - 1)$. Hence as the network utilization level α gets closer to the utilization bound $1/(H^* - 1)$, the worst-case delay bound approaches rapidly to infinity.

The elegant result of Charny and Le Boudec raises several interesting and important questions regarding the design and provisioning power of aggregate packet scheduling. In this paper we will take a more theoretical perspective and attempt to address the fundamental trade-offs in the design of aggregate packet scheduling algorithms and their provisioning power in support of (worst-case) guaranteed delay service. In particular, we study the relationships between the worst-case edge-to-edge delay (i.e., the maximum delay experienced by any packet across a network domain), the maximum allowable network utilization level, and the "sophistication/complexity" of aggregate packet scheduling employed by a network. *À la* the Internet DiffServ paradigm, we consider a framework where user traffic is only conditioned (i.e., shaped) at the edge of a network domain, whereas inside the network core, packets are scheduled based solely on certain bits (referred to as the *packet state* carried in the packet header). In other words, the aggregate packet scheduling algorithm employed inside the network core maintains no per-flow/user information, thus it is *core-stateless*.

In our framework, besides the conventional "Type of Service (TOS)" or "Differentiated Service (DS)" bits, we assume that additional control information may be carried in the packet header for scheduling purpose. By encoding certain *timing* information in the packet header, we design two classes of aggregate packet scheduling algorithms: the *static earliest time first* (SETF) and *dynamic earliest time first* (DETF) algorithms. In the class of SETF packet scheduling algorithms, packets are stamped with its entry time at the network edge, and they are scheduled in the order of their time stamps (i.e., their network entry times) inside the network core; the class of DETF packet scheduling algorithms work in a similar fashion, albeit with an important difference—the packet time stamps are updated at certain routers (hence the term *dynamic*). In both classes, the granularity of timing information encoded in the packet state—as is determined by the number of bits used for packet state encoding—is a critical factor that affects the provisioning power of aggregate packet scheduling.

The contribution of our study is that, using these two classes (SETF and DETF) of aggregate packet scheduling algorithms, in addition to the simple FIFO discipline, we explore the fundamental trade-offs

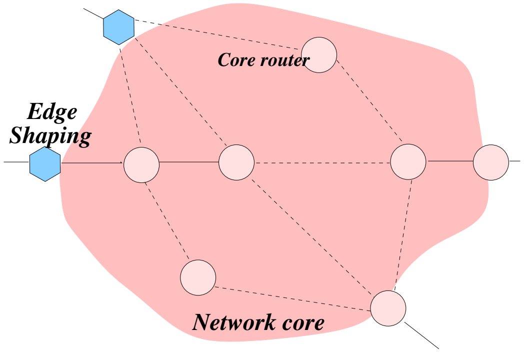

Fig. 1. The network model.

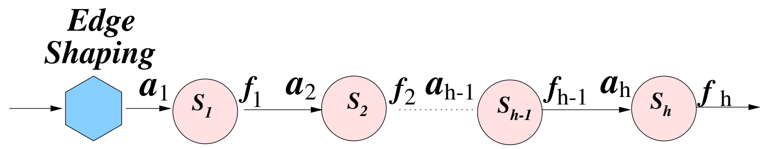

Fig. 2. Packet's arrival time at and departure time from each scheduler.

in aggregate packet scheduling:

- how, with additional control information encoded in the packet state and with added "sophistication/complexity" in aggregate packet scheduling, the worst-case edge-to-edge delay bound and the maximum allowable network utilization bound can be improved;
- how these performance bounds are affected by the number of bits available for packet state encoding.

Through analysis and numerical examples, we show that when packet time stamps are encoded with the finest time granularity, i.e., carrying the precise releasing time of the packets, both the SETF and DETF packet scheduling algorithms can attain an arbitrary network utilization level (i.e., α can be arbitrarily close to 1). In other words, the maximum allowable network utilization bound is independent of the network diameter H^*. This is in contrast to the case of FIFO, where the known maximum utilization level is bounded by $1/(H^* - 1)$. Furthermore, using the more complex DETF, the worst-case edge-to-edge delay bound is linear in H^*, whereas using the simpler SETF, the worst-case edge-to-edge delay bound is inversely proportional to $(1-\alpha)^{H^*}$. When packet time stamps are encoded using coarser granularity (i.e., the number of bits for packet state encoding is limited), the network utilization level is constrained by the time granularity. In addition, the worst-case edge-to-edge delay bound is increased. With the same number of bits, the more complex DETF packet scheduling algorithms have far superior performance over the simpler SETF algorithms.

The remainder of the paper is organized as follows. In Section II we present the basic model and assumptions for our analysis. The two classes of aggregate packet scheduling, SETF and DETF, are analyzed and the trade-offs discussed in Section III and Section IV, respectively. We conclude the paper in Section V.

II. Network Model and Assumptions

Before we start, it is worth noting that all the major notations used in this paper are summarized in Table I.

H^*	Network diameter
α	Network utilization level
β	Traffic burstiness
(r_S, e_S)	Rate and latency parameters of a GR server S
e	$e = \max_{all S's}\{e_S\}$
Δ	maximum transmission time of an arbitrary packet
Γ	Duration of a time slot
h^*	Maximum number of hops that a packet can traverse within Γ (in SETF(Γ)); Hop count when a packet's time stamp needs to be updated (in DETF)
D_{H^*}	Worst-case end-to-end delay bound
m	Number of bits needed to encode time stamp information
C^*	Maximum link capacity in a network
ι	$\iota = 1/C^*$, finest time granularity
d^*	Time stamp increment in DETF

TABLE I
NOTATION USED IN THIS PAPER

Consider a single network domain, as shown in Figure 1, where we distinguish network edge routers from core routers. We assume that a number of traffic classes (or aggregates) are supported by the network domain. Conceptually, we can imagine that there is a separate queue for each traffic class at a core router. We further assume that for each traffic aggregate, the router provides a Guaranteed Rate (GR) service curve (also called rate-latency service curve) [4] to the aggregate. In particular, if a router provides a rate-latency service curve $\delta(t) = r_d(t - e_d)^+$ to a traffic aggregate, we would say that for the traffic aggregate, the router is a GR node with parameters (r_d, e_d). We refer to r_d and e_d as the reserved rate and latency for the aggregate at the router, respectively. Note that, this definition of GR node is independent of the kind of scheduling discipline used for scheduling packets *within* the traffic aggregate. We call this scheduling discipline for packets within the aggregate as aggregate packet scheduling discipline and assume that, for a traffic aggregate, all routers employ the same aggregate packet scheduling algorithm (e.g., FIFO) that performs packet scheduling using only certain bits (the *packet state*) carried in the packet header. No other scheduling information is used or stored at core routers. We refer to the aggregate scheduling mechanism employed at an outgoing link of a router[1] as a *scheduler*. For clarity, we may incorporate the name of aggregate packet scheduling discipline into the notion of the GR node. For example, if the FIFO discipline is used for a traffic aggregate and the aggregate receives a GR service curve with parameters (r_d, e_d) at a router, then we would refer to this router as a GR-FIFO node (scheduler) with parameters (r_d, e_d) (or simply FIFO node with parameters (r_d, e_d) when there is no confusion) for the traffic aggregate. In this paper, we only focus on the aggregate packet scheduling discipline for *a single* traffic aggregate. Without loss of generality, the remaining discussions in the paper are all related to this single traffic aggregate (e.g. an EF aggregate). Moreover, we assume that all traffic of the aggregate entering the network is

[1]For simplicity, we assume that output-queueing is used.

shaped at the edge traffic conditioner before releasing into the network. No traffic shaping or re-shaping is performed inside the network core.

Consider a GR scheduler S with parameters (r_S, e_S). We denote the MTU (maximum transmission unit) of the link by L_S^{max}, then $\Delta_S = L_S^{max}/r_S$ is the transmission time of an MTU-sized packet with the reserved rate r_S. Define $\Delta = \max_{allS's}\{\Delta_S\}$, and $e = \max_{allS's}\{e_S\}$. We also assume that the path of any user flow is pre-determined, and fixed throughout its duration. Let H^* be the maximum number of hops in the paths that any user flow may traverse in the network. We refer to H^* as the *network diameter*.

Consider an arbitrary flow j (of the aggregate) traversing the network. The traffic of the flow is shaped at the network edge in such a manner that it conforms to a *token bucket regulated arrival curve* (σ^j, ρ^j) [7]: Let $A^j(t, t+\tau)$ denote the amount of the flow j traffic released into the network during a time interval $[t, t+\tau]$, where $t \geq 0$, $\tau \geq 0$; then $A^j(t, t+\tau) \leq \sigma^j + \rho^j\tau$. We control the overall network utilization level *by imposing a utilization factor* α on each link as follows (with respect to the reserved rate of the aggregate). Consider an arbitrary GR scheduler S with parameters (r_S, e_S). Let $\mathcal{F}$ denote the set of user flows traversing S in the aggregate. Then the following condition holds:

$$\sum_{j \in \mathcal{F}} \rho^j \leq \alpha r_S. \tag{1}$$

where $0 < \alpha \leq 1$. We will also refer to the utilization factor α as the *network utilization level* of a network domain. In addition to the link utilization factor α, we will also impose an overall bound $\beta \geq 0$ (in units of time) on the "burstiness" of flows traversing scheduler S: $\sum_{j \in \mathcal{F}} \sigma^j \leq \beta r_S$. As we will see later, this *burstiness factor* β plays a less critical role in our analysis than the network utilization level α. It is worth noting that both the network utilization level α and the burstiness factor β are defined with respect to the *reserved rate* of an aggregate instead of link capacities.

From the above edge shaping and network utilization constraints, we can obtain an important bound on the amount of traffic of the aggregate going through a given scheduler *that is injected at the network edge during any time interval*. Consider an arbitrary GR scheduler S with parameters (r_S, e_S). For any time interval $[\tau, t]$, let $\mathring{A}_S(\tau, t)$ denote the amount of traffic of the aggregate injected into the network during the time interval $[\tau, t]$ that will traverse S (at perhaps some later time). Here we use $\mathring{A}$ to emphasize that $\mathring{A}_S(\tau, t)$ is *not* the traffic traversing S during the time interval $[\tau, t]$, *but* injected into the network at the network edge during $[\tau, t]$. Using the facts that $A^j(t, t+\tau) \leq \sigma^j + \rho^j\tau$ for all flows, $\sum_{j \in \mathcal{F}} \rho^j \leq \alpha r_S$ and $\sum_{j \in \mathcal{F}} \sigma^j \leq \beta r_S$, it is easy to show that

$$\mathring{A}_S(\tau, t) \leq \alpha r_S(t - \tau) + \beta r_S. \tag{2}$$

We refer to this bound as the *edge traffic provisioning condition* for scheduler S. As we will see later, the edge traffic provisioning condition is critical to our analysis of aggregate packet scheduling algorithms.

Now consider a packet p (of any flow) that traverses a path with $h^p \leq H^*$ hops. For $i = 1, 2, \ldots, h^p$, denote the scheduler at the ith hop on the path of packet p as S_i (see Figure 2). Let a_i^p and f_i^p represent, respectively, the time that packet p arrives at and departs[2] from scheduler S_i. For ease of exposition, throughout this paper we assume that the propagation delay from one scheduler to another scheduler is zero. Hence $a_{i+1}^p = f_i^p$. Note that a_1^p is the time packet p is released into the network (after going through the edge traffic conditioner). Define $d_i^p = f_i^p - a_1^p$, i.e. d_i^p is the delay experienced by the packet after traversing the ith hop. (Note that the delay experienced by a packet at the edge traffic conditioner is excluded from the edge-to-edge delay.) Hence $d_{h^p}^p$ is the cumulative delay that packet p experiences along its path, and is referred to as the *edge-to-edge* delay experienced by packet p. Define D_i to be the worst-case edge-to-edge delay experienced by any packet in the network after traversing i hops,

$$D_i = \max_{\text{all } p\text{'s with } h^p \geq i} \{d_i^p\}, . \tag{3}$$

Therefore D_{H^*} is the worst-case edge-to-edge delay experienced by any packet in the aggregate in the network.

The key questions that we will address in the remainder of the paper are: 1) given an aggregate packet scheduling algorithm, under what network utilization level α does an upper bound on D_{H^*} exist? 2) how does this bound depend on the network utilization level α and the network diameter H^*? and 3) how these relationships are affected by the number of bits available for packet state encoding as well as the added “sophistication/complexity” in aggregate packet scheduling?

In order to compare with the new aggregate packet scheduling disciplines that we will study in this paper, and to illustrate the trade-offs in designing aggregate scheduling, we restate the performance bounds of a general FIFO network by Charny and Le Boudec here [6], using the above set of notation, before we leave this section.

Theorem 1: Given a network of FIFO schedulers with a network diameter H^*, if the network utilization level α satisfies the condition $\alpha < \frac{1}{H^*-1}$, then the worst-case edge-to-edge delay D_{H^*} is bounded above by

$$D_{H^*} \leq \frac{H^*(e+\beta)}{1-(H^*-1)\alpha}. \tag{4}$$

III. Network of Static Earliest Time First Schedulers

In this section we will design and analyze a new class of aggregate packet scheduling algorithms—the class of *static earliest time first* (SETF) algorithms. Using this class of aggregate packet scheduling

[2]Throughout the paper we adopt the following convention: a packet is considered to have arrived at a scheduler *only* when its last bit has been received, and it to have departed from the scheduler *only* when its last bit has been serviced.

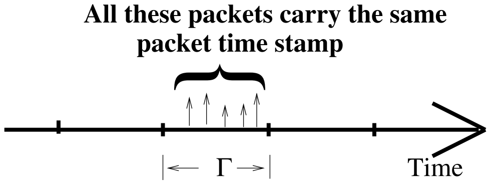

Fig. 3. Time slots and packet time stamps.

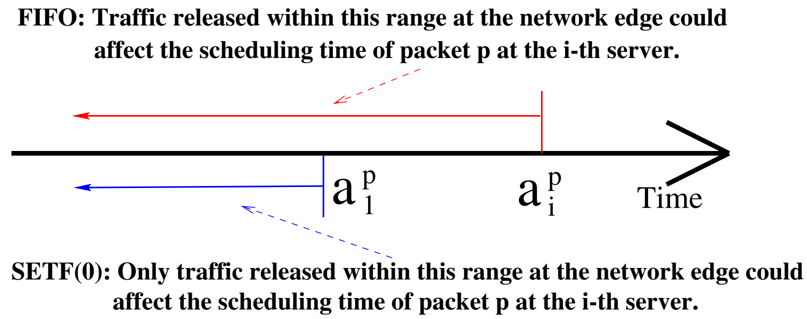

Fig. 4. Illustration of the different behaviors of FIFO and SETF(0).

algorithms, we will demonstrate how by adding some "sophistication/complexity" in aggregate packet scheduling—in particular, by encoding additional control information in the packet header, we can improve the maximum allowable utilization bound, and reduce the provable worst-case edge-to-edge delay bound, compared with the FIFO scheduling discipline. Furthermore, we will discuss the performance trade-offs of SETF packet algorithms when a limited number of bits is used for packet state encoding.

The additional control information used by the class of SETF schedulers is a *(static) time stamp* carried in the packet header of a packet that records the time the packet is released into the network (after going through the edge traffic conditioner) at the network edge. Here we assume that all edge devices that time-stamp the packets use a global clock (in other words, the clocks at the edge devices are synchronized). We denote the time stamp of a packet p by ω_0^p. An SETF scheduler inside the network core schedules packets in the order of their time stamps, ω_0^p. Note that in the case of SETF, the time stamp of a packet is never modified by any SETF scheduler, thus the term *static*.

Depending on the time granularity used to represent the packet time stamps, we can design a class of SETF schedulers with different performance/complexity trade-offs. We use SETF(Γ) to denote the SETF packet scheduling algorithm where packet time stamps are represented with time granularity Γ. In particular, SETF(0) denotes the SETF packet scheduling algorithm where packet time stamps are represented with the *finest* time granularity, namely, packets are time-stamped with the *precise* time they are released into the network. Formally, for any packet p, we have $\omega_0^p = a_1^p$. For a more general SETF(Γ) scheduling algorithm where $\Gamma > 0$, we divide the time into slots of Γ time units each (see Figure 3): $t_n = [(n-1)\Gamma, n\Gamma), n = 1, 2, \ldots$. Packets released into the network are time-stamped with the corresponding time slot number n. In other words, packets that are released into the network within the same time slot (say, the time slot $t_n = [(n-1)\Gamma, n\Gamma)$) carry the same time stamp value, i.e., $\omega_0^p = n$. Therefore, packets released into the network during the same time slot at the network edge are *indistinguishable* by an SETF(Γ) scheduler inside the network core, and are serviced by the scheduler in a FIFO manner. We will show later that using coarser time granularity (i.e., larger Γ) can potentially reduce the number of bits needed to encode the packet time stamps, but at the expenses of degrading the performance bounds.

In the rest of this section, we first establish the performance bounds for SETF, and then discuss the packet state encoding issue. At the end of this section, we conduct numerical studies to illustrate the performance trade-offs and provisioning power of SETF.

A. Performance Bounds for a Network of SETF Schedulers

Note that SETF(0) can be considered as a special case of SETF(Γ), where $\Gamma = 0$. Therefore, we focus on deriving the performance bounds for a network of SETF(Γ). The bounds for SETF(0) are presented thereafter as a corollary. Consider a network of SETF(Γ) schedulers. Recall that under SETF(Γ), the time is divided into time slots and packets released into the network during the same time slot carry the same time stamp value (i.e., the time slot number). Clearly the coarser the time granularity Γ is, more packets will be time-stamped with the same time slot number. In particular, if Γ is larger than the worst-case edge-to-edge delay of the network, then a network of SETF(Γ) schedulers degenerates to a network of FIFO schedulers.

Before we present the performance bounds for a network of SETF(Γ) schedulers, we first introduce a new notation, h^*: for a given Γ, define $h^* + 1$ to be the maximum number of hops that any packet can reach within Γ units of time after it is released into the network. Mathematically, h^* is the smallest h such that the following relation holds for all packets:

$$\min_{\text{all } p\text{'s}} \{a^p_{h^*+1} - a^p_1\} \geq \Gamma. \tag{5}$$

The definition h^* also presents an intuitive guideline for the amount of traffic that may interfere with each other. Consider an arbitrary packet p and arbitrary GR-SETF(Γ) scheduler S along its path, then we know that, a packet entering the network in the time interval $[a^p_{h^*+1}, a^p_S]$ must have a time stamp that is larger than that of packet p. Therefore, largely speaking, these packet will not interfere the scheduling of packet p at scheduler S. On the other hand, for $i \leq h^*$, packets entering the network after a^p_1 but before a^p_S may have the same time stamp as packet p and compete with packet p for service. Note that if $h^* = 0$, we must have $\Gamma = 0$. This gives us SETF(0). On the other hand, if Γ is large enough such that $h^* = H^* - 1$, SETF(Γ) becomes FIFO.

We state the performance bounds for a network of GR-SETF(Γ) schedulers in the following theorem. For the clarity of exposition, its proof is delegated in the Appendix .

Theorem 2: Consider a network of GR-SETF(Γ) schedulers with a network diameter H^*. If the network utilization level α satisfies the following condition,

$$(1-\alpha)^{H^*-h^*-1} > \alpha h^*, \tag{6}$$

then the worst-case edge-to-edge delay is bounded above by,

$$D_{H^*} \le \frac{(\beta+e)h^* + \alpha^{-1}(\beta+e+\Delta)\{1-(1-\alpha)^{H^*-h^*}\}}{(1-\alpha)^{H^*-h^*-1} - \alpha h^*}. \tag{7}$$

By setting $h^* = 0$ in Theorem 2, we obtain the performance bounds for a network of SETF(0) schedulers (see Theorem 3 below), whereas letting $h^* = H^* - 1$, we have the results for a network of FIFO schedulers (with a difference of $\frac{\Delta}{1-(H^*-1)\alpha}$ caused by the extra care taken by the analysis of an SETF network to account for the non-preemptive property of an SETF scheduler, see Theorem 1 in Section II).

Theorem 3: Consider a network of GR-SETF(0) schedulers with a network diameter H^*. The worst-case edge-to-edge delay D_{H^*} is bounded above by,

$$D_{H^*} \le \frac{\alpha^{-1}(\beta+e+\Delta)\{1-(1-\alpha)^{H^*}\}}{(1-\alpha)^{H^*-1}},\ 0 < \alpha < 1. \tag{8}$$

Comparing with a network of FIFO schedulers, we see that in a network of SETF(0) schedulers, the network utilization level can be kept as high (i.e., as close to 1) as wanted: unlike FIFO, there is no limit on the maximum allowable network utilization level. However, since the worst-case edge-to-edge delay bound is inversely proportional to $(1-\alpha)^{H^*-1}$, it increases exponentially as $\alpha \to 1$. On the other hand, in general, Theorem 2 states that with a coarser time granularity $\Gamma > 0$ (which determines h^*), we may no longer be able to set the network utilization level at any arbitrary level, as in the case of SETF(0), while still having a finite worst-case edge-to-edge delay bound. In other words, for a given $\Gamma > 0$, there is a limit on the maximum allowable network utilization level as imposed by the condition (6). This limit on the maximum allowable network utilization level is the performance penalty we pay for using coarser time granularity to represent the packet time stamp information. We will conduct numerical studies in Section III-C to illustrate these performance trade-offs.

B. Time Stamp Encoding

In this section we discuss the implication of the worst-case edge-to-edge delay bound on the number of bits needed to encode the time stamp information. Again we consider a network of SETF(Γ) first. Suppose that m bits are sufficient to encode the packet time stamps precisely. Then the time-stamp bit string wraps around every $2^m\Gamma$ units of time. Given that the worst-case edge-to-edge delay of a packet in the network of SETF(Γ) is bounded above by D_{H^*}, we must have $2D_{H^*} \le 2^m\Gamma$ so as to enable any SETF(Γ) scheduler to correctly distinguish and compare the time stamps of two different packets[3]. From

[3]Here we assume that no extra clock or other timing device/mechanism is used to assist an SETF(Γ) scheduler to distinguish the packet time stamps. In other words, an SETF(Γ) scheduler must use the bit strings encoded in the packet header to determine whether the time stamp of one packet is smaller than that of another packet. This can be achieved, for example, by using the *lollipop sequence number* technique [10]. Note that if we assume that each SETF(Γ) scheduler has a clock that is synchronized with the edge time stamping devices, and thus can use this clock to identify the time slot the current time corresponds to, then it is sufficient to have $2^m\Gamma \ge D^*$, i.e., one less bit is needed in this case.

Theorem 2, we have

$$m \geq \log_2\{\frac{(\beta+e)h^* + \alpha^{-1}(\beta+e+\Delta)\{1-(1-\alpha)^{H^*-h^*}\}}{((1-\alpha)^{H^*-h^*-1} - \alpha h^*)\Gamma}\} + 1. \tag{9}$$

From (9), we see that for a fixed network utilization level α, larger Γ may reduce the number of bits needed for packet time stamp encoding. However, as we increase Γ, h^* may also be increased. Consequently, the right hand side of (9) may increase. Hence the relationship between m and Γ is not strictly monotone. Furthermore, a larger Γ in general also yields a smaller maximum allowable network utilization level bound.

Now let us derive the number of bits needed to encode the time stamp information in a network of SETF(0) schedulers. Suppose that C^* is the maximum link capacity of the network. Then it is sufficient to have a time granularity of $\iota = 1/C^*$ to mark the precise time each bit of data enters the network[4]. In other words, $\iota = 1/C^*$ is the finest time granularity needed to represent packet time stamps. In the remainder of this paper we will assume that the clock granularity of the edge devices that place time stamps on packets entering the network is at least ι, i.e., the clocks tick (at least) every ι units of time. Following the same argument as above and from Theorem 3, we have the number of bits, m, that is needed to encode the time stamp information in SETF(0),

$$m \geq \log_2\{\frac{\alpha^{-1}(\beta+e+\Delta)\{1-(1-\alpha)^{H^*}\}}{((1-\alpha)^{H^*-1})\iota}\} + 1. \tag{10}$$

C. Numerical Studies

In this section we perform numerical studies to illustrate the performance trade-offs and provisioning power of SETF schedulers. In all the studies, we assume that the capacity of all links is $10\,Gb/s$, and all packets have the same size $L = 1000$ bytes. Moreover, we assume that there is only one traffic aggregate, i.e. all the routers are GR nodes for the aggregate with parameters $(10\,Gb/s, 0)$ (we make such an assumption in all the following numerical examples). We set the *network burstiness factor* β in a similar manner as in [6]: we assume that the token bucket size of each flow is bounded in such a way that $\sigma^j \leq \beta_0\rho^j$, where β_0 (measured in units of time) is a constant for all flows. For a given network utilization level α, we then set $\beta = \alpha\beta_0$. In all the numerical studies presented in this paper, we choose $\beta_0 = 25\,ms$.

Figure 5 compares the worst-case edge-to-edge bounds for a FIFO network and an SETF(0) network (with $H^* = 8$) as a function of the network utilization level α. From Figure 5, it is clear that for a given

[4]Although theoretically speaking the finest time granularity $\Gamma = 0$, it is obvious that in practice $\iota = 1/C^*$ is sufficient, as no two bits can arrive at any link within ι units of time.

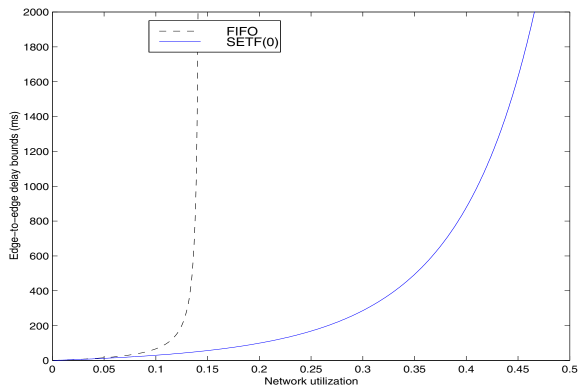

Fig. 5. Performance comparison: SETF(0) vs. FIFO.

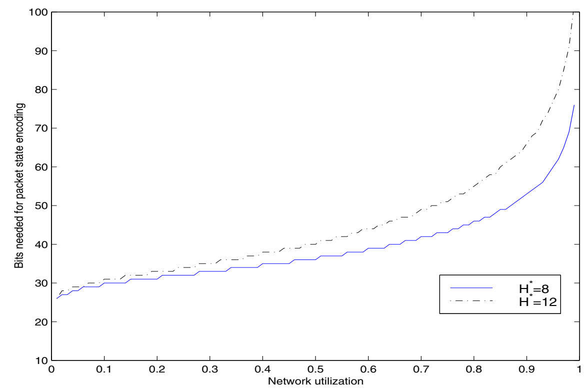

Fig. 6. No. of bits needed for encoding for SETF(0).

network utilization level, the worst-case edge-to-edge delay bound for an SETF(0) network is much better than that for a FIFO network. On the other hand, because SETF(0) needs to encode a fine-grained time stamp information in packet headers, we expect that the cost in terms of the number of bits needed to encode time stamp will be high, as illustrated in Figure 6. This figure shows the number of bits needed for packet time stamp encoding for two SETF(0) networks with $H^* = 8$ and $H^* = 12$, respectively. The other parameters used in this example are the same as in Figure 5. In particular, $C^* = 10Gb/s$, and thus $\iota = 1/C^* = 10^{-7}\,ms$. As expected, the number of bits needed for packet time stamp encoding increases as the network utilization level increases; it also increases as the network diameter scales up. From this figure we also see that even for a relative low network utilization level, the number of bits required for packet time stamp encoding is relatively large. For example, with $H^* = 8$, 26 bits are needed for $\alpha = 0.1$. Consequently, to achieve a meaningful network utilization level, an SETF(0) network requires a large number of bits for packet time stamp encoding, thus incurring significant control overhead. Below we conduct numerical studies on SETF(Γ) to show how this problem can be potentially addressed by using coarser time granularity for packet time stamp encoding in SETF(Γ). First we note that from Theorem 2, (9) and the definition of h^* (5), we can see that given a network with diameter H^*, we can essentially divide the time granularity Γ into H^* *granularity levels*: each granularity level corresponds to one value of $h^* = 0, 1, \ldots, H^* - 1$. The *finest* granularity level corresponds to $h^* = 0$, and the *coarsest* granularity level to $h^* = H^* - 1$. For this reason, in the following numerical studies, we will use h^* to indicate the time granularity used in an SETF(Γ) network.

Figure 7 shows the effect of time granularity on the worst-case edge-to-edge delay bound for an SETF(Γ) network with $H^* = 8$. For comparison, we also include the results for the corresponding FIFO network. From the figure it is clear that coarser time granularity (i.e., larger h^*) yields poorer worst-case edge-to-edge delay bound. As the time granularity gets coarser (i.e., h^* increases), the worst-case edge-to-edge delay bound quickly approaches to that of the FIFO network.

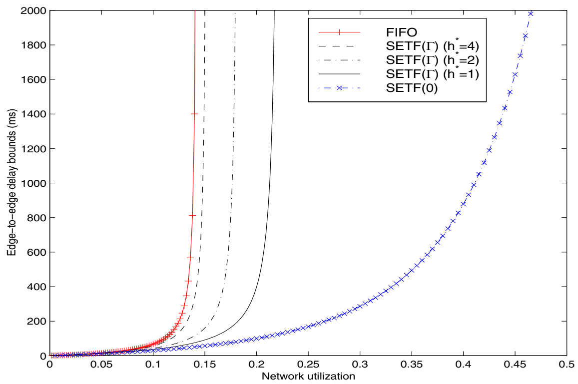

Fig. 7. Performance comparison: SETF vs. FIFO.

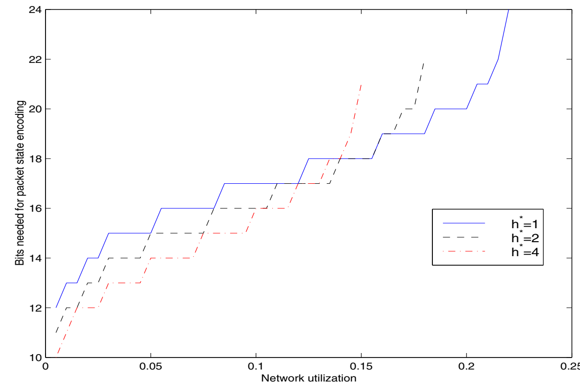

Fig. 8. No. of bits needed for encoding for SETF(Γ).

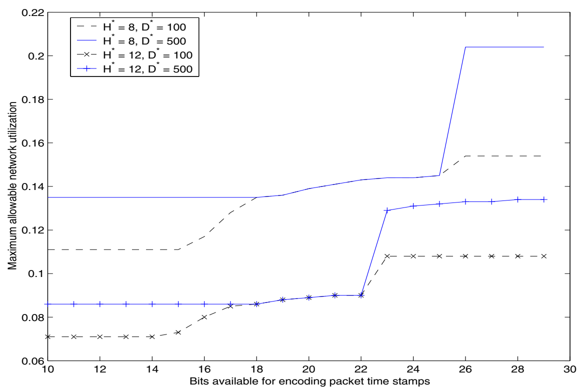

Fig. 9. No. of bits for encoding, network diameter, and maximum allowable network utilization.

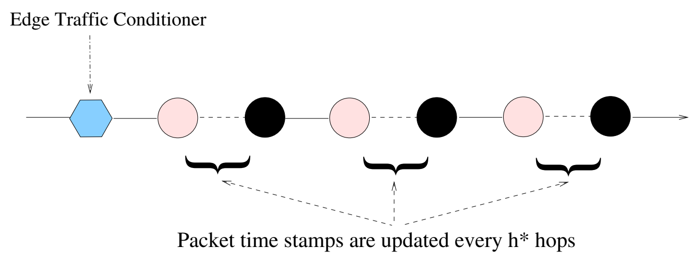

Fig. 10. Updating packet time stamps inside the network core.

Next we illustrate how the network utilization level of an SETF(Γ) network affects the number of bits needed for packet time stamp encoding. Figure 8 shows the number of bits needed for packet time stamp encoding as a function of the network utilization level under various time granularities (as indicated by h^*). In this example, the network diameter $H^* = 8$. From this figure we see that for low network utilization levels, using coarser time granularity reduces the number of bits needed for packet time stamp encoding. However, as coarser time granularity also imposes a tight bound on the maximum allowable network utilization, this reduction in the number of bits needed for packet time stamp encoding *may not be feasible* when the network utilization level is increased (this is why the curve for a given time granularity (h^*) stops at certain network utilization level). To put it in another way, to achieve a higher network utilization level, SETF(Γ) schedulers with *finer* time granularity (thus smaller h^*) must be used, thus requiring more bits for packet time stamp encoding.

In the last set of numerical studies, we demonstrate how the number of bits available for packet time stamp encoding affects the maximum allowable network utilization so as to support a given target worst-case edge-to-edge delay bound for SETF networks. The results are shown in Figure 9, where networks with a combination of the network diameters $H^* = 8$ and $H^* = 12$ and delay bounds $D_{H^*} = 100\,ms$

and $D_{H^*} = 500\,ms$ are used. As we can see from the figure that for a given number of bits for packet time stamp encoding, as the network diameter increases, the maximum allowable network utilization decreases. Note also that when the number of bits for packet time stamp encoding is small (e.g., less than 15 for a network with parameters $H^* = 8$ and $D_{H^*} = 100\,ms$), the packet time stamp does not enhance the performance of a SETF(Γ, h^*) network, and the SETF(Γ, h^*) network behaves essentially as a FIFO network with a maximum network utilization level around 0.11. Beyond this threshold, as the number of bits used increases, the maximum allowable network utilization also increases. However, as the figure shows, further increasing the number of bits beyond a certain value (e.g., 26 for a network with parameters $H^* = 8$ and $D^* = 100\,ms$) for encoding will not improve the maximum allowable network utilization.

IV. Network of Dynamic Earliest Time First Schedulers

So far we have seen that by including additional control information in the packet header and adding sophistication/complexity at network schedulers, the class of SETF packet scheduling algorithms improve upon the maximum allowable network utilization and worst-case edge-to-edge delay bounds of the simple FIFO packet scheduling algorithm. This performance improvement comes essentially from the ability of an SETF scheduler to limit the effect of "newer" packets on "older" packets. However, the provisioning power of SETF packet scheduling algorithms is still rather limited. Given the finest time granularity to encode the packet time stamps, although we can achieve arbitrary network utilization in a network of SETF(0) schedulers, the worst-case edge-to-edge delay bound is inversely proportional to $(1-\alpha)^{H^*}$. Hence the bound grows exponentially, as the network diameter H^* increases. In addition, with coarser time granularities, the performance of SETF networks deteriorates further. In this section we devise another class of aggregate packet scheduling algorithms—the class of DETF algorithms—which with further "sophistication/complexity" added at the schedulers, achieve far superior performance.

In the general definition of a DETF packet scheduling algorithm, we use two parameters: the time granularity Γ and the (packet) time stamp increment hop count h^*. Note that unlike SETF where h^* is determined by Γ, here h^* is independent of Γ. Hence we denote a DETF scheduler by DETF(Γ, h^*). In the following, we will present the definition of DETF($0, h^*$) first, i.e., DETF with the finest time granularity. The general definition of DETF(Γ, h^*) will be given afterwards.

As in the case of SETF(0), the time stamp of a packet in a network of DETF($0, h^*$) schedulers is represented precisely. In particular, it is initialized at the network edge with the time the packet is released into the network. Unlike SETF(0), however, the time stamp of the packet will be updated every h^* hops (see Figure 10). Formally, suppose packet p traverses a path of h hops. Let ω_0^p denote the time stamp of packet p as it is released into the network, i.e., $\omega_0^p = a_1^p$. Let $\kappa = \lceil \frac{h}{h^*} \rceil$. For $k = 1, 2, \ldots, \kappa - 1$, the time stamp of packet p is updated after it has traversed the kh^*th hop on its path (or as it enters the (kh^*+1)th hop on its path). Let ω_k^p denote the packet time stamp of packet p after its kth update. The packet time stamp ω_k^p is updated using the following update rule:

$$\omega_k^p := \omega_{k-1}^p + d^*, \qquad k = 1, \dots, \kappa - 1, \tag{11}$$

where the parameter $d^* > 0$ is referred as the (packet) time stamp increment. We impose the following condition on d^* that relates the packet time stamp ω_k^p to the actual time packet p departs the kh^*th hop:

$$\text{for } k = 1, \dots, \kappa - 1, \quad f_{kh^*}^p \leq \omega_k^p, \text{ and } f_h^p \leq \omega_\kappa^p := \omega_{\kappa-1}^p + d^*. \tag{12}$$

This condition on d^* is referred to as the *reality check* condition. Intuitively, we can think of the path of packet p being partitioned into κ *segments* of h^* hops each (except for the last segment, which may be shorter than h^* hops). The reality check condition (12) ensures that the packet time stamp carried by packet p after it has traversed k segments is not smaller that the actual time it takes to traverse those segments. In the next section we will see that the reality check condition (12) and the packet time stamp update rule (11) are essential in establishing the performance bounds for a network of DETF schedulers.

We now present the definition for the general DETF(Γ, h^*) packet scheduling algorithm with a (coarser) time granularity $\Gamma > 0$. As in the case of SETF(Γ), in a network of DETF(Γ, h^*) schedulers, the time is divided into time slots of Γ units: $[(n-1)\Gamma, n\Gamma), n = 1, 2, \dots$, and all packet time stamps are represented using the time slots. In particular, if packet p is released into the network in the time slot $[(n-1)\Gamma, n\Gamma)$, then $\omega_0^p = n\Gamma$. We also require that the packet time stamp increment d^* be a multiple of Γ. Hence the packet time stamp ω_k^p is always a multiple of Γ. In practice, we can encode ω_k^p as the corresponding time slot number (as in the case of SETF(Γ)).

A. Performance Bounds for a Network of DETF Schedulers

In this section we establish performance bounds for a network of DETF schedulers. In particular, we will show that by using dynamic packet time stamps, we can obtain significantly better performance bounds for a network of DETF schedulers than those for a network of SETF schedulers.

Consider a network of DETF(Γ, h^*) schedulers, where $\Gamma \geq 0$ and $1 \leq h^* \leq H^*$. We first establish an important lemma which bounds the amount of traffic carried by packets at a DETF(Γ, h^*) scheduler whose time stamp values fall within a given time interval. Consider a DETF(Γ, h^*) scheduler S. Given a time interval $[\tau, t]$, let $\mathcal{M}$ be the set of packets that traverse S at some time whose time stamp values fall within $[\tau, t]$. Namely, $p \in \mathcal{M}$ if and only if for some $k = 1, 2, \dots, \kappa,$, S is on the kth segment of packet p's path, and $\tau \leq \omega_{k-1}^p \leq t$. For any $p \in \mathcal{M}$, we say that packet p *virtually* arrives at S during $[\tau, t]$. Let $\tilde{A}_S(\tau, t)$ denote the total amount of traffic virtually arriving at S during $[\tau, t]$, i.e., total amount of traffic carried by packets in $\mathcal{M}$. Then we have the following bound on $\tilde{A}_S(\tau, t)$.

Lemma 4: Consider an arbitrary GR scheduler S with parameters (r_S, e_S) in a network of DETF(Γ, h^*) schedulers. For any time interval $[\tau, t]$, let $\tilde{A}(\tau, t)$ be defined as above. Then

$$\tilde{A}(\tau, t) \leq \beta r_S + \alpha r_S (t - \tau + \Gamma). \tag{13}$$

Proof: For simplicity, we first prove a bound on $\tilde{A}^j(\tau, t)$, the amount of traffic virtually arriving at S during $[\tau, t]$ from a flow j. Consider an arbitrary packet p of flow j which virtually arrives at S (on the kth segment) during $[\tau, t]$, i.e., $\tau \leq \omega_{k-1}^p \leq t$. From (11), it is easy to see that,

$$\omega_{k-1}^p = \omega_0^p + (k-1)d^*.$$

Because $\tau \leq \omega_{k-1}^p \leq t$, we have,

$$\tau - (k-1)d^* \leq \omega_0^p \leq t - (k-1)d^*.$$

Therefore,

$$\tilde{A}^j(\tau, t) \leq \sigma^j + \rho^j \lceil \frac{t - (k-1)d^* - (\tau - (k-1)d^*)}{\Gamma} \rceil \Gamma \leq \sigma^j + \rho^j (t - \tau + \Gamma). \tag{14}$$

From (14) and the edge traffic provisioning condition (2), the lemma follows easily. ■

Note that if $\Gamma = 0$, the bound on $\tilde{A}(\tau, t)$ is exactly the same as the edge traffic provisioning condition (2). Intuitively, (13) means that using the (dynamic) packet time stamp with the finest time granularity, the amount of traffic *virtually* arriving at S during $[\tau, t]$ is bounded in a manner as if the traffic were re-shaped at S using (2). In the general case where a coarser time granularity $\Gamma > 0$ is used, an extra $\alpha r_S \Gamma$ amount of traffic may (virtually) arrive at S, as opposed to (2) at the network edge. This is not surprising, since with a coarser time granularity, a scheduler S inside the network core cannot distinguish a packet from those other packets that traverse S and have the same time stamp value.

From Lemma 4, we can derive a recursive relation for ω_k^p's using a similar argument as used before. Based on this recursive relation, we can establish performance bounds for a network of DETF(Γ, h^*) schedulers. The general results are somewhat "messy" to state. For brevity, in the following we present results for three special but *representative* cases. As we will see later, the first two theorems are sufficient to demonstrate the provisioning power of a network of DETF schedulers. The third theorem is included here for comparison purpose. Their proofs can be found in [13].

Theorem 5 (A Network of GR-DETF(0,1) Schedulers): Consider a network of GR-DETF(0,1) schedulers with a network diameter H^*. Let $d^* = \beta + e + \Delta$, then the reality condition (12) holds. Furthermore, for any $0 < \alpha < 1$, the worst-case edge-to-edge delay D^* is bounded above by $D^* \leq H^* d^* = H^*(\beta + e + \Delta)$.

Theorem 6 (A Network of GR-DETF($\Gamma, 1$) Schedulers): Consider a network of GR-DETF($\Gamma, 1$) schedulers with a network diameter H^*, where $\Gamma > 0$. Let $d^* = \lceil (\alpha\Gamma + \beta + e + \Delta)/\Gamma \rceil \Gamma$, then the reality condition (12) holds. Furthermore, for any $0 < \alpha < 1$, the worst-case edge-to-edge delay D^* is bounded above by $D^* \leq H^* d^* + \Gamma$.

Theorem 7 (A Network of GR-DETF(Γ, h^*) Schedulers with $d^* = \Gamma$): Consider a network of GR-DETF(Γ, h^*) schedulers with a network diameter H^*, where $\Gamma > 0$ (and $h^* > 1$). We set $d^* = \Gamma$, *i.e., the packet time stamp is advanced exactly one time slot every time it is updated.* Let $\kappa^* = \lceil \frac{H^*}{h^*} \rceil$. Suppose the network utilization level α and the time granularity Γ satisfy the following condition:

$$0 < \frac{h^*(\alpha\Gamma + \beta + e + \Delta)}{1 - (h^* - 1)\alpha} \leq d^* = \Gamma. \tag{15}$$

Then the worst-case edge-to-edge delay D^* is bounded above by $D^* \leq (\kappa^* + 1)\Gamma$.

From Theorem 5 and Theorem 6, we see that with $h^* = 1$, the worst-case edge-to-edge delay bound is linear in the network diameter H^*. Furthermore, with the finest time granularity, the worst-case edge-to-edge delay bound is independent of the network utilization level α. This is because the per-hop delay is bounded by $d^* = \beta + e + \Delta$. With a coarser time granularity $\Gamma > 0$, per-hop delay is bounded by $d^* = \lceil (\alpha\Gamma + \beta + e + \Delta)/\Gamma \rceil \Gamma$, where the network utilization level determines the "additional delay" ($\alpha\Gamma$) that a packet may experience at each hop.

From Theorem 7, we see that in a network of DETF(Γ, h^*) where $d^* = \Gamma$ and $h^* > 1$, the maximum allowable network utilization is bounded. To see why this is the case, first note that we must have $\alpha < 1/(h^* - 1)$, otherwise the left hand side of (15) becomes infinity. For a given $\Gamma > 0$, the condition (15) imposes the following tighter bound on α:

$$\alpha < \frac{1 - h^*(\beta + e + \Delta)\Gamma^{-1}}{2h^* - 1} < \frac{1}{2h^* - 1} < \frac{1}{h^* - 1}. \tag{16}$$

For a given α that satisfies (16), comparing the worst-case edge-to-edge delay bound in Theorem 7 to that of a network of FIFO schedulers with a network diameter h^*, we see that updating packet time stamps every h^* hops effectively reduces a network of diameter H^* into a number of smaller networks with diameter h^*. In particular, setting $d^* = \Gamma$ allows us to consider these smaller networks as networks of FIFO schedulers with diameter h^*. By appropriately taking into account the effect of dynamic time stamps with coarser time granularity (the extra $\alpha\Gamma + \Delta$ factor), Theorem 7 can essentially be obtained from the bound for a network of FIFO schedulers.

B. Packet State Encoding

In this section we first consider the problem of packet state encoding for a network of DETF schedulers, namely, the number of bits that is needed to encode the *dynamic* packet time stamp and possibly other control information for the proper operation of a DETF network.

First consider a network of DETF(0,1) schedulers with a network diameter H^*. As in the case of SETF(0), we use ι to denote the finest time granularity necessary to represent the packet time stamps, i.e., $\iota = 1/C^*$, where C^* is the maximum link capacity of the network. From Theorem 5, we see that the number of bits m that is needed to encode the (dynamic) packet time stamps precisely must satisfy the following condition:

$$2^{m-1}\iota \geq H^*(\beta + e + \Delta), \text{ or } m \geq \log_2 H^* + \log_2[(\beta + e + \Delta)/\iota] + 1. \tag{17}$$

Now consider a network of DETF($\Gamma, 1$) with a coarser time granularity $\Gamma > 0$. From Theorem 6, for a given network utilization level α, we see that the number of bits m that is needed to encode the (dynamic) packet time stamps must satisfy the following condition:

$$2^{m-1}\Gamma \geq H^*\lceil\frac{\alpha\Gamma + \beta + e + \Delta}{\Gamma}\rceil\Gamma + \Gamma, \text{ or } m \geq \log_2\{H^*\lceil\frac{\alpha\Gamma + \beta + e + \Delta}{\Gamma}\rceil + 1\} + 1. \tag{18}$$

Hence for a given network utilization level α, coarser time granularity (i.e., larger Γ) in general leads to fewer bits needed to encode the dynamic packet time stamps. However, due to the ceiling operation in (18), at least $\log_2\{H^*+1\}+1$ bits are needed. This *effectively* places a bound on the range of time granularities that should be used, i.e., $\Gamma \in [0, (\beta+e+\Delta)/(1-\alpha)]$. Any coarser time granularity $\Gamma > (\beta+e+\Delta)/(1-\alpha)$ will not reduce the minimum number of bits, $\log_2\{H^* + 1\} + 1$, needed for packet time stamp encoding.

In the general case where $h^* > 1$, in order to ensure a DETF(Γ, h^*) scheduler to work properly, not only do we need to encode the packet time stamps, we also need some additional control information to be carried in the packet header of each packet: in order for a scheduler to know whether the packet time stamp of a packet must be updated, we include a hop-count counter as part of the packet state carried in the packet header to record the number of hops a packet has traversed. This hop-count counter is incremented every time a packet traverses a scheduler, and it is reset when it reaches h^*. Thus the hop-count counter can be encoded using $\log_2 h^*$ number of bits. Therefore for a network of DETF(Γ, h^*) where d^* is set to Γ, from Theorem 7 the total number of bits needed for packet state encoding is given by

$$m \geq \log_2\{\kappa^* + 1\} + 1 + \log_2 h^*, \tag{19}$$

provided that the network utilization level α and the time granularity Γ are chosen in such a manner that (15) holds. Note that from (19) we have $m \geq \log_2\{\kappa^* h^* + h^*\} + 1 \geq \log_2\{H^* + h^*\} + 1$. Therefore, the number of bits needed for encoding the packet states is increased as h^* increases. Moreover, via (15) h^* also affects the maximum allowable network utilization bound. In particular, from (16) a larger h^* leads to a smaller bound on the maximum allowable network utilization. For these reasons it is sufficient to

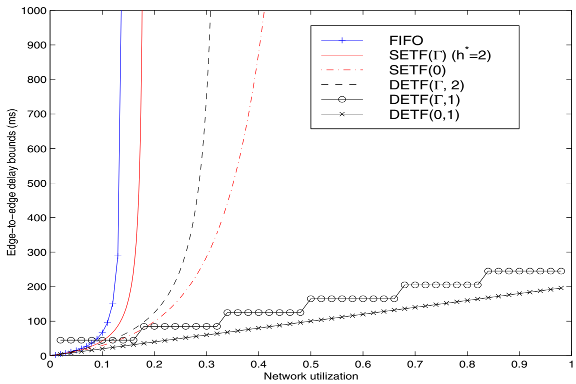

Fig. 11. Edge-to-edge delay bound comparison ($H^* = 8$).

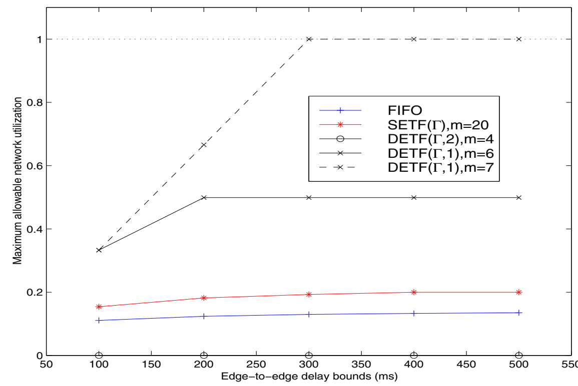

Fig. 12. Provisioning power of FIFO, SETF(Γ), DETF($\Gamma, 1$), and DETF($\Gamma, 2$) networks ($H^* = 8$).

only consider networks of DETF($\Gamma, 1$) schedulers[5].

C. Performance Trade-offs and Provisioning Power of Aggregate Packet Scheduling

In this section we use numerical examples to demonstrate the performance trade-offs in the design of DETF networks. By comparing the performance of FIFO, SETF and DETF networks, we also illustrate the provisioning power of the aggregate packet scheduling algorithms in support of guaranteed delay service. Lastly, we briefly touch on the issue of complexity/cost in implementing the aggregate packet scheduling algorithms.

The network setting for all the studies is the same as before. Namely, all links have a capacity of $10\,Gb/s$, all packets have a size of $L = 1000\,B$, and $\beta = \alpha\beta_0$, where α is the network utilization level and $\beta_0 = 25\,ms$. Moreover, as before, only one traffic aggregate is supported by the network for simplicity. The network diameter H^* and the network utilization level α will be varied in different studies.

In the first set of numerical examples, we illustrate the relationship between the network utilization level α and the worst-case edge-to-edge delay bound for networks employing various aggregate packet scheduling algorithms. The results are shown in Figure 11, where $H^* = 8$ is used for all the networks. For the SETF(Γ) network, we choose $\Gamma = 2\Delta = 0.8\mu s$ (i.e., $h^* = 2$). Whereas in the DETF($\Gamma, 2$) network, the time granularity Γ is chosen in such a way that (15) in Theorem 7 holds. For the DETF($\Gamma, 1$) network, we set $\Gamma = 5\,ms$. From the figure we see that the DETF(0,1) network has the best worst-case edge-to-edge delay bound. Despite a relatively coarser time granularity, the delay bound for the DETF($\Gamma, 1$) network is fairly close to that of the DETF(0,1) network. In addition, when the network utilization level is larger than 0.2, the DETF($\Gamma, 1$) network also has a better delay bound than the rest of the networks. From

[5]In practice, it is possible to implement the hop-count counter using, say, the TTL field in the IP header, thus avoiding the extra $\log_2 h^*$ bits. For example, we have implemented two versions of DETF packet scheduling algorithms in FreeBSD: one using IP-IP tunneling technique; another using MPLS. In both cases, we only need additional bits to encode the packet time stamps. In such situations, a network of DETF(Γ, h^*) schedulers with $d^* = \Gamma > 0$ and $h^* > 1$ requires only $\log_2 \kappa^* + 1$ additional number of bits.

Theorem 6, it is clear that the worst-case edge-to-edge delay bound for a DETF($\Gamma, 1$) network decreases (and approaches to that of a DETF($0, 1$) network), when finer time granularity (smaller Γ) is used. The delay bound of the DETF($\Gamma, 2$) network is worse than that of the SETF(0) network (with the finest time granularity), but is considerably better than those of the SETF(Γ) and FIFO networks. From this example, we see that the DETF networks in general have far better delay performance than those of SETF and FIFO networks.

In the next set of numerical examples, we compare the provisioning power of the various aggregate packet scheduling algorithms. In particular, we consider the following provisioning problem: given a network employing a certain aggregate packet scheduling algorithm, what is the maximum allowable network utilization level we can attain in order to meet a target worst-case edge-to-edge delay bound? In this study, we allow networks employing different aggregate packet scheduling algorithms to use different number bits for packet state encoding. More specifically, the FIFO network needs no additional bits. The SETF(Γ) network (where Γ is chosen such that $h^* = 1$) uses 20 additional bits for time stamp encoding. The number of additional bits used by the DETF($\Gamma, 2$) network is 3. For the DETF($\Gamma, 1$) networks, we consider two cases: one uses 6 additional bits, while the other uses 7 bits. All the networks used in these studies have the same diameter $H^* = 8$. Figure 12 shows the maximum allowable network utilization level as a function of the target worst-case edge-to-edge delay bound for the various networks. The results clearly demonstrate the performance advantage of the DETF networks. In particular, with a few number of bits needed for packet state encoding, the DETF($\Gamma, 1$) networks can attain much higher network utilization level, while supporting the same worst-case edge-to-edge delay bound.

In the last set of numerical examples, we focus on the DETF($\Gamma, 1$) networks only. In this study, we investigate the design and performance trade-offs in employing DETF($\Gamma, 1$) networks to support guaranteed delay service. In particular, we consider the following problem: given a fixed number of bits for packet state encoding, what is the maximum allowable network utilization level that we can attain to support a target worst-case edge-to-edge delay bound? Note that for a network of diameter H^*, at least $\log_2\{H^*+1\}+1$ bits are needed for packet state encoding. More bits available will allow us to choose finer time granularity for time stamp encoding, thus yielding a better delay bound as well as a higher maximum network utilization level. In Figure 13 we show, for a network of diameter $H^* = 8$, how the number of bits available for packet state encoding affects the maximum network utilization level so as to support a given target worst-case edge-to-edge delay bound. The same results for a network of diameter $H^* = 12$ are shown in Figure 14. From these results we see that with a relatively few number of bits, a DETF network can achieve fairly decent or good network utilization while meeting the target worst-case edge-to-edge delay bound. In particular, with the target worst-case edge-to-edge delay bounds $200\,ms$ and $500\,ms$, we can achieve more than 50% (and up to 100%) network utilization level using only 6 to 7 additional

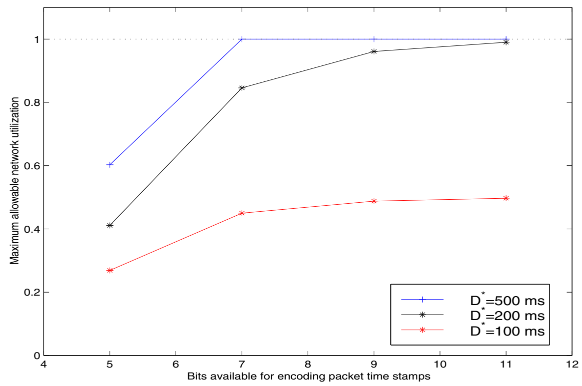

Fig. 13. Design and performance trade-offs for DETF($\Gamma, 1$) networks ($H^* = 8$).

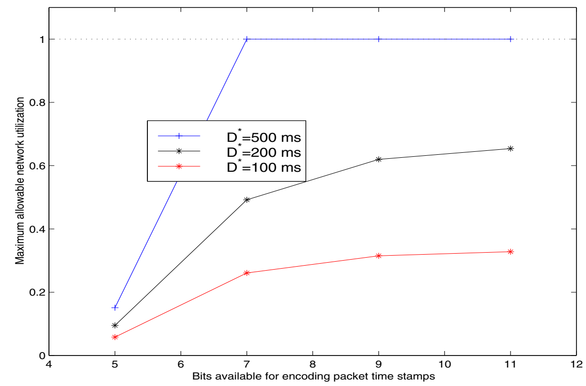

Fig. 14. Design and performance trade-offs for DETF($\Gamma, 1$) networks ($H^* = 12$).

bits. Comparing Figure 13 and Figure 14, it is clear that a network with larger diameter requires more bits than a network with smaller diameter to achieve the same maximum allowable network utilization. However, the minimum number of bits required for packet state encoding grows only logarithmically with the network diameter H^*. Furthermore, today's networks tend to be more "dense", i.e., with relative small H^*. Hence with relatively small number of additional bits (e.g., 8 or 16 bits) for time stamp encoding, we can design DETF($\Gamma, 1$) networks to attain fairly high network utilization while supporting reasonably good edge-to-edge delay bounds.

We conclude this section by briefly touching on the issue of cost/complexity in implementing the aggregate packet scheduling algorithms. Besides the fact that additional bits are needed for packet state encoding, both the SETF and DETF packet scheduling algorithms require comparing packet time stamps and sorting packets accordingly. With the finest time granularity, this sorting operation can be expensive. However, with only a few bits used for packet time stamp encoding, sorting can be avoided by implementing a "calendar queue" (or rotating priority queue [11]) with a number of FIFO queues. This particularly favors the DETF($\Gamma, 1$) packet scheduling algorithms, since the number of bits needed for time stamp encoding can be kept small. However, compared to SETF, DETF($\Gamma, 1$) packet scheduling algorithms require updating packet time stamps at every router, and thus d^* must be configured at each router. Lastly, in terms of finding additional bits for packet state encoding, we can re-use certain bits in the IP header [12]. This is the case in our prototype implementation using the IP-IP tunneling technique, where we re-use the IP identification field (16 bits) in the encapsulating IP header to encode the packet time stamp.

V. Conclusions and Future Work

In this paper we investigated the fundamental trade-offs in aggregate packet scheduling for support of (worst-case) guaranteed delay service. Based on a novel analytic approach that focuses on network-wide performance issues, we studied the relationships between the worst-case edge-to-edge delay, the maximum allowable network utilization level and the "sophistication/complexity" of aggregate packet scheduling employed by a network. We designed two new classes of aggregate packet scheduling algorithms—the static earliest time first (SETF) and dynamic earliest time first (DETF) algorithms—both of which employ additional timing information carried in the packet header for packet scheduling, but differ in their manipulation of the packet time stamps. Using the SETF and DETF as well as the simple FIFO packet scheduling algorithms, we demonstrated that with additional control information carried in the packet header and added "sophistication/complexity" at network schedulers, both the maximum allowable network utilization level and the worst-case edge-to-edge delay bound can be significantly improved. We further investigated the impact of the number of bits available for packet state encoding on the performance trade-offs as well as the provisioning power of these aggregate packet scheduling algorithms. In particular, we showed that with relatively small number of bits for packet state encoding, the DETF packet scheduling algorithms can attain fairly good performance bounds. These results illustrate the fundamental trade-offs in the design of aggregate packet scheduling algorithms, and shed light on the provisioning power of aggregate packet scheduling in support of guaranteed delay service.

There are a number of research directions we are currently exploring. By taking into account the actual network topology, we are extending the analytic approach presented in this paper to obtain better performance bounds. Such analysis may also help us identify "hot spots" and "bottleneck" links in a network, and therefore allow us to possibly make special provisioning for network "hot spots" and "bottleneck" links. Using the insights obtained in this paper, we are also studying the stochastic traffic behavior in a network with aggregate packet scheduling. Through this study we hope to obtain useful provisioning rules for providing predictable Internet QoS services (e.g., along the line of [3]). Extensions to the DETF aggregate packet scheduling for supporting multiple delay classes and rate guarantees are also under investigation.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers of IEEE ICNP'2001 and the Transactions on Parallel and Distributed Systems for many valuable comments. This work was supported in part by NSF Grants CAREER Award NCR-9734428, EIA-9818338, and ITR ANI-0085824. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] J. Bennett, K. Benson, A. Charny, W. Courtney, and J.-Y. Le Boudec. Delay jitter bounds and packet scale rate guarantee for expedited forwarding. In *Proc. IEEE INFOCOM*, Anchorage, AL, April 2001.

[2] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An architecture for differentiated services. RFC 2475, December 1998.

[3] T. Bonald, A. Proutiere, and J. Roberts. Statistical guarantees for streaming flows using expedited forwarding. In *Proc. IEEE INFOCOM*, Anchorage, AL, April 2001.

[4] J.-Y. Le Boudec and P. Thiran. *Network Calculus*. Springer-Verlag, July 2001. available online at http://lcawww.epfl.ch.

[5] C.-S. Chang. *A Filtering Theory for Communication Networks*. Springer-Verlag, 1999.

[6] A. Charny and J.-Y. Le Boudec. Delay bounds in a network with aggregate scheduling. In *Proceedings of QoFIS*, Berlin, Germany, October 2000.

[7] R. Cruz. A calculus for network delay, Part I: Network elements in isolation. *IEEE Transactions on Information Theory*, 37(1):114–131, January 1991.

[8] V. Jacobson, K. Nichols, and K. Poduri. An expedited forwarding PHB. RFC 2598, June 1999.

[9] V. Jacobson, K. Nichols, and K. Poduri. The ‘virtual wire’ per-domain behavior. Internet Draft, July 2000. Work in Progress.

[10] S. Keshav. *An Engineering Approach to Computer Networking*. Addison-Wesley, 1997.

[11] J. Liebeherr and D. E. Wrege. A versatile packet multiplexer for quality-of-service networks. In *Proc. 4th International Symposium on High Performance Distributed Computing (HPDC-4)*, pages 148–155, August 1995.

[12] I. Stoica and H. Zhang. Providing guaranteed services without per flow management. In *Proc. ACM SIGCOMM*, Boston, MA, September 1999.

[13] Z.-L. Zhang, Z. Duan, and Y. T. Hou. Fundamental trade-offs in aggregate packet scheduling. Technical report, Computer Science Department, University of Minnesota, February 2001. Available at http://www.cs.umn.edu/˜zhzhang/papers.html.

APPENDIX

Proof of Theorem 2: The proof consists of two parts. In the first part we show that if a finite bound on the edge-to-edge delay exists, then the upper bound given by (7) holds. In the second part, we show that a finite bound exists.

Part I: We assume that a finite bound on the edge-to-edge delay exists. Consequently, a finite bound on D_{H^*-1} exists. Consider an arbitrary packet p that traverses a path with h hops. Let S_i be the ith hop along the path. Then we have

$$f_i^p \leq e_i + a_i^p + \frac{\mathcal{M}(a_i^p, f_i^p)}{r_i} \tag{20}$$

where r_i and e_i are the reserved rate and latency parameters at the GR node for the aggregate traffic, respectively. We consider two cases of index i: $1 \leq i \leq h^*$ and $i \geq h^* + 1$.

case 1 ($1 \leq i \leq h^*$)**:** for this case, it is possible that all the packets in the system have a timestamp that is less than or equal to ω_i^p, therefore all the traffic serviced during the time interval $(a_i^p, f_i^p]$ should have been released from the network edge between $[a_i^p - D_{H^*-1}, a_i^p]$. Hence

$$\begin{aligned} f_i^p - a_i^p &\leq e_i + \frac{\AA_i(a_i^p - D_{H^*-1}, a_i^p)}{r_i} \\ &\leq e + \alpha D_{H^*-1} + \beta, \end{aligned} \tag{21}$$

hence

$$d_i \le i(\alpha D_{H^*-1} + \beta + e). \tag{22}$$

Case 2 ($i \ge h^* + 1$)**:** from the definition of h^*, we know that a packet entering the network in the time interval $[h^p_{h^*+1}, a^p_i]$ must have a timestamp that is larger than that of packet p. Therefore all the traffic serviced during the time interval $(a^p_i, f^p_i]$ must have been released into the network edge between $[a^p_i - D_{H^*}, a^p_{h^*+1}]$ (except the packet (if it exists) being serviced at time a^p_i, which may have an releasing time ω_0 larger than $a^p_{h^*+1}$. Note that this packet has a size of at most L^{max}_i).

$$\begin{aligned} f^p_i &\le e_i + a^p_i + \frac{L^{max}_i + \mathring{A}_i(a^p_i - D_{H^*-1}, a^p_{h^*+1})}{r_i} \\ &\le e + a^p_i + \alpha(D_{H^*-1} - (a^p_i - a_{h^*+1})) + \beta + \Delta. \end{aligned} \tag{23}$$

Therefore,

$$f^p_i - a^p_i \le \alpha(D_{H^*-1} - (a^p_i - a^p_1) + (a_{h^*+1} - a^p_1)) + \beta + e + \Delta \tag{24}$$

Hence

$$d^p_{i+1} \le d^p_i + \alpha(D_{H^*-1} - d^p_i) + \alpha h^*(\alpha D_{H^*-1} + \beta + e) + \beta + e + \Delta. \tag{25}$$

Let $d^p_0 = 0$ and by iteration we have, for $i = 1, 2, \ldots, h^*$,

$$d^p_i \le i(\alpha D_{H^*-1} + \beta + e); \tag{26}$$

and for $i = h^* + 1, \ldots, h$,

$$d^p_i \le h^*(\alpha D_{H^*-1} + \beta + e) + D_{H^*-1}\{1 - (1-\alpha)^{i-h^*}\} + \alpha^{-1}(\beta + e + \Delta)\{1 - (1-\alpha)^{i-h^*}\}. \tag{27}$$

Without loss of generality, let $i = H^* - 1$, we obtain that if the network utilization level α satisfies the following condition,

$$(1-\alpha)^{H^*-h^*-1} > \alpha h^* \tag{28}$$

then

$$D_{H^*-1} \le \frac{(\beta+e)h^* + \alpha^{-1}(\beta+e+\Delta)\{1-(1-\alpha)^{H^*-h^*-1}\}}{(1-\alpha)^{H^*-h^*-1} - \alpha h^*}. \tag{29}$$

Letting $i = H^*$ and plugging in the bound on D_{H^*-1}, we have

$$D_{H^*} \le \frac{(\beta+e)h^* + \alpha^{-1}(\beta+e+\Delta)\{1-(1-\alpha)^{H^*-h^*}\}}{(1-\alpha)^{H^*-h^*-1} - \alpha h^*}. \tag{30}$$

Part II: We follow the same argument as in [6] to show that a finite bound on the edge-to-edge delay indeed exists. In particular, we adopt the time-stopping approach here [5]. In this approach, we assume that all the traffic sources can simultaneously stop at any instant of time t, we then examine the state of the system under this condition. We refer to this system as a virtual system, which are indexed by the time instant when all the traffic sources stop. Consider an arbitrary time $t > 0$ and a virtual system made of the original network, where all sources stop sending traffic at time t. It can be proven that, under the condition $(1-\alpha)^{H^*-h^*-1} > \alpha h^*$, the amount of traffic buffered at all the schedulers is finite (we omit the proof here). Let $D(t)$ denote the worst case edge-to-edge delay experienced by any packet in the virtual system indexed by time t. It is clear that $D(t)$ is bounded, given that all schedulers only have finite amount of queued traffic, and the sources stop sending traffic. Let t tend to infinity, we see that the worst case delay remains bounded. ∎

PLACE PHOTO HERE

Zhenhai Duan (S '97–M '03/ACM M '03) received the B.S. degree from Shandong University, China, in 1994, the M.S. degree from Beijing University, China, in 1997, and the Ph.D. degree from the University of Minnesota, in 2003, all in Computer Science. He is currently an Assistant Professor in the Computer Science Department at the Florida State University. His research interests include computer networks and multimedia communications, especially the scalable network resource control and management in the Internet, Internet routing protocols and service architectures, and networking security. Dr. Duan is a co-recipient of the 2002 IEEE International Conference on Network Protocols Best (ICNP) Paper Award. He is a member of IEEE and ACM.

PLACE
PHOTO
HERE

Zhi-Li Zhang (M '97/ACM M '97) received the B.S. degree in computer science from Nanjing University, China, in 1986 and his M.S. and Ph.D. degrees in computer science from the University of Massachusetts in 1992 and 1997. In 1997 he joined the Computer Science and Engineering faculty at the University of Minnesota, where he is currently an Associate Professor. From 1987 to 1990, he conducted research in Computer Science Department at Århus University, Denmark, under a fellowship from the Chinese National Committee for Education. He has held visiting positions at Sprint Advanced Technology Labs; IBM T.J. Watson Research Center; Fujitsu Labs of America, Microsoft Research China, and INRIA, Sophia-Antipolis, France. His research interests include computer communication and networks, especially QoS, routing and security issues in Internet, multimedia and real-time systems, and modeling and performance evaluation of computer and communication systems.

Dr. Zhang currently serves on the Editorial board of *IEEE/ACM Transactions on Networking* and *Computer Network, an International Journal*. He is Technical Program Co-chair of IEEE INFOCOM 2006 and the IEEE/IFIP IWQoS'04 as well as the SPIE ITCOM 2002 conference on Scalability and Traffic Control in IP Networks, served on the Executive Committee for IEEE Infocom 2001 and Infocom 2003, and on the Technical Program Committees of various conferences and workshops including IEEE Infocom, IEEE ICNP, ACM SIGCOMM, ACM SIGMETRICS and ACM SIGMM.

He received the National Science Foundation CAREER Award in 1997. He has also awarded the prestigious McKnight Land-Grant Professorship at the University of Minnesota, and the Miller Visiting Professorship at Miller Institute for Basic Sciences, University of California, Berkeley. Dr. Zhang is co-recipient of an ACM SIGMETRICS best paper award and an IEEE Internation Conference on Network Protocols (ICNP) best paper award. He is a member of IEEE, ACM and INFORMS Telecommunication Section.

PLACE
PHOTO
HERE

Yiwei Thomas Hou (S '91–M '98–SM '04 / ACM M '95) obtained his B.E. degree from the City College of New York in 1991, the M.S. degree from Columbia University in 1993, and the Ph.D. degree from Polytechnic University, Brooklyn, New York, in 1998, all in Electrical Engineering. From 1997 to 2002, Dr. Hou was a principal research scientist and project leader at Fujitsu Laboratories of America, IP Networking Research Department, Sunnyvale, California (Silicon Valley). Since Fall 2002, he has been an Assistant Professor at Virginia Tech, the Bradley Department of Electrical and Computer Engineering, Blacksburg, Virginia. Tom's research interests are in the algorithmic design and optimization for network systems. His current research focuses on wireless ad hoc networks, sensor networks, and video over ad hoc networks. In recent years, he has worked on scalable architectures, protocols, and implementations for differentiated services Internet; service overlay networking; multimedia streaming over the Internet; and network bandwidth allocation policies and distributed flow control algorithms. He has published over 100 journal and conference papers in the above areas and is a co-recipient of the 2002 IEEE International Conference on Network Protocols (ICNP) Best Paper Award and the 2001 IEEE Transactions on Circuits and Systems for Video Technology (CSVT) Best Paper Award. He is a member of ACM and a senior member of IEEE.