Fundamental Trade-offs in Aggregate Packet Scheduling

Zhi-Li Zhang†, Zhenhai Duan† and Yiwei Thomas Hou‡

† Dept. of Computer Science & Engineering
University of Minnesota
Minneapolis, MN 55455
{zhzhang,duan}@cs.umn.edu

‡ Fujitsu Labs of America
595 Lawrence Expressway
Sunnyvale, CA 94085
thou@fla.fujitsu.com

Abstract

In this paper we investigate the fundamental trade-offs in *aggregate* packet scheduling for support of guaranteed delay service. In our study, besides the simple FIFO packet scheduling algorithm, we consider two new 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, 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 better 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.

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) [6]. 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 [7]—has been questioned [1], [3].

In a recent work by Charny and Le Boudec [3], it is shown that in order to provide guaranteed delay service using FIFO, the overall *network utilization level* must be limited to a small fraction of its link capacities. More specifically, in a network of FIFO schedulers, the *worst-case* delay at each router is bounded 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 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 "TOS" 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 new 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 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 objective of our study is to use these two new classes (SETF and DETF) of aggregate packet scheduling algorithms, in addition to the simple FIFO discipline, to illustrate the fundamental trade-offs in aggregate packet scheduling: 1) *how*

1092-1658/01 $17.00 © 2001 IEEE

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; and 2) *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, 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 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. In Section III, we re-establish the result in [3] using our approach. The two new classes of aggregate packet scheduling, SETF and DETF, are analyzed and the trade-offs discussed in Section IV and Section V, respectively. We conclude the paper in Section VI.

II. Network Model and Assumptions

Consider a single network domain, as shown in Figure 1, where all traffic entering the network is shaped at the edge traffic conditioner before releasing into the network. No traffic shaping or re-shaping is performed inside the network core. We assume that 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 scheduling mechanism employed at an outgoing link of a router as a *scheduler*. Let C be the capacity of the corresponding outgoing link of a scheduler S. We will also refer to C as the capacity of the scheduler S. We denote the MTU (maximum transmission unit) of the link by L^{max}, then L^{max}/C is the transmission time of an MTU-sized packet. Define $\Delta = \max_{all S's}\{L^{max}/C\}$, i.e., Δ is the maximum transmission time of any packet in the network. We assume that the path of any user flow is predetermined, 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 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) [4]: 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. Consider an arbitrary scheduler S with capacity C. Let $\mathcal{F}$ denote the set of user flows traversing S. Then the following condition holds:

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

Clearly, $0 < \alpha \leq 1$. 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 any scheduler S: $\sum_{j\in\mathcal{F}} \sigma^j \leq \beta C$. As we will see later, this *burstiness factor* β plays a less critical role in our analysis than the network utilization level α.

From the above edge shaping and network utilization constraints, we can obtain an important bound on the amount of traffic going through a given scheduler *that is injected at the network edge during any time interval*. Consider an arbitrary scheduler S with capacity C. For any time interval $[\tau, t]$, let $\mathring{A}_S(\tau, t)$ denote the amount of traffic 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, $\sum_{j\in\mathcal{F}} \rho^j \leq \alpha C$ and $\sum_{j\in\mathcal{F}} \sigma^j \leq \beta C$, it is easy to show that

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

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 $h \leq H^*$ hops. For $i = 1, 2, \ldots, h$, 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 departsfrom 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), and f_h^p is the time packet p leaves the network. Hence $f_h^p - a_1^p$ 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. (Note that the delay experienced by a packet at the edge traffic conditioner is excluded from the edge-to-edge delay.) Define D^* to be the worst-case edge-to-edge delay experienced by any packet in the network, i.e.,

$$D^* = \max_{\text{all } p\text{'s}}\{f_h^p - a_1^p\}, \tag{3}$$

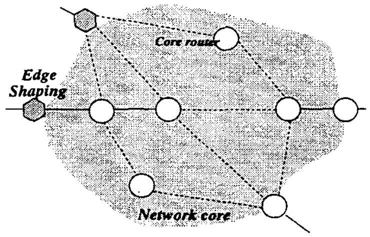

Fig. 1. The network model.

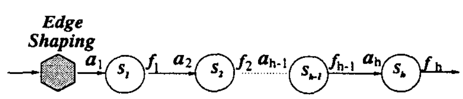

Fig. 2. Packet's arrival time at and departure time from each scheduler.

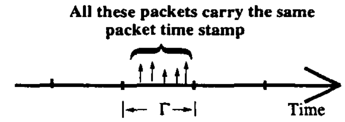

Fig. 3. Time slots and packet time stamps.

where in the above definition h is the number of hops on the path of packet p.

The key questions we will address in the remainder of the paper are: 1) given an aggregate scheduling algorithm, under what network utilization level α does an upper bound on D^* exist? 2) how does this bound depend on the utilization level α and 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?

III. Network of FIFO Schedulers

In this section we re-establish the result of Charny and Le Boudec [3] for a network of FIFO schedulers *using a different approach*. Unlike [3] which uses an argument based on the worst-case per-hop delay analysis, in our approach we attempt to obtain a recursive relation for a_i^p's (or equivalently, f_i^p's) for any packet p. From this recursive relation we then derive an upper bound on the worst-case edge-to-edge delay D^*. As we will see later, this argument is quite general and powerful, and forms the basis of all the analyses in this paper.

A key step in our analysis is to obtain an upper bound on the amount of traffic that is serviced by a scheduler between the arrival and departure of any packet p at the scheduler. This bound will allow us to establish a recursive relation between a_{i+1}^p and a_i^p. For this purpose, we introduce an important notation, τ^*, which is *the maximum time it takes for any packet to reach its last hop*. Formally,

$$\tau^* = \max_{\text{all } p\text{'s}}\{a_h^p - a_1^p\}. \tag{4}$$

Now consider a FIFO scheduler S of capacity C. Let a_S^p denote the time a packet p arrives at S, and f_S^p the time packet p departs from S. Define $Q(a_S^p)$ to be *the amount of traffic serviced by the scheduler S between* $[a_S^p, f_S^p]$. Note that since S is a FIFO scheduler, $Q(a_S^p)$ is exactly the amount of traffic queued at S at the arrival time of packet p (with packet p itself included). We have the following bound on $Q(a_S^p)$:

Lemma 1: For a FIFO scheduler S of capacity C, we have

$$Q(a_S^p) \le \alpha C\tau^* + \beta C. \tag{5}$$

Proof: Let p^* be the *last* packet before packet p (itself inclusive) that when packet p^* arrives at scheduler S any packet p' in the queue (including the one in service) satisfies the following condition:

$$a_1^{p'} \ge a_1^{p*}. \tag{6}$$

In other words, when packet p^* arrives at scheduler S, it is the "oldest" packet in the queue: namely, *all other packets currently in the queue entered the network no early than packet* p^*. We note that such a packet always exists—if no other packets satisfy (6), the packet that starts the current busy period certainly does. Let $a_S^{p^*}$ denote the time packet p^* arrived at scheduler S. By the definition of p^*, any packet that was either queued at scheduler S at time $a_S^{p^*}$ or arrived at scheduler S between $a_S^{p^*}$ and a_S^p must have entered the network during the time interval $[a_1^{p^*}, a_S^p]$. From (2), the amount of traffic carried by these packets is bounded above by $\alpha C(a_S^p - a_1^{p^*}) + \beta C$. Furthermore, since scheduler S is always busy during $[a_S^{p^*}, a_S^p]$, we have

$$Q(a_S^p) \le \alpha C(a_S^p - a_1^{p^*}) + \beta C - (a_S^p - a_S^{p^*})C. \tag{7}$$

As $a_S^p - a_1^{p^*} = a_S^p - a_S^{p^*} + a_S^{p^*} - a_1^{p^*}$ and $a_S^{p^*} - a_1^{p^*} \le \tau^*$, from (7) we see that (5) follows easily. ■

There is an intuitive explanation of the result in Lemma 1. Note that a FIFO scheduler services packets in the order of their arrival times at the scheduler, regardless of when they are released into the network. In particular, packets entering the network later than packet p can potentially be serviced earlier than packet p. Intuitively, packets that are queued at the time packet p arrives at scheduler S must have entered the network between $[a_S^p - \tau^*, a_S^p]$ and arrived at scheduler S before packet p. By the edge traffic provisioning condition (2), the amount of traffic carried by these packets is bounded by $\alpha C\tau^* + \beta C$. This intuitive argument is made rigorous in the proof of Lemma 1.

We now use Lemma 1 to derive a recursive relation for a_i^p's. Consider a packet p which traverses a path with h hops. The capacity of the ith scheduler on the path is denoted by C_i. Then by the definition of $Q(a_i^p)$, we have

$$a_{i+1}^p = f_i^p = a_i^p + Q(a_i^p)/C_i \le a_i^p + \alpha\tau^* + \beta. \tag{8}$$

Recursively applying (8) and using the relation $f_i^p = a_{i+1}^p$, we have the following lemma.

Lemma 2: Consider a packet p which traverses a path with h hops. Then, for $i = 1, 2, \ldots, h$, we have,

$$f_i^p - a_1^p \le i(\alpha\tau^* + \beta). \tag{9}$$

Using Lemma 2, we can establish the following main results for a network of FIFO schedulers[1].

Theorem 3: 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 $\tau^* \leq \frac{(H^*-1)\beta}{1-(H^*-1)\alpha}$. Furthermore, the worst-case edge-to-edge delay D^* is bounded above by

$$D^* \leq \frac{H^*\beta}{1-(H^*-1)\alpha}. \quad (10)$$

Theorem 3 illustrates the provisioning power of a network of FIFO schedulers for support of guaranteed delay service: in order to provide a *provable* worst-case edge-to-edge delay bound, the maximum network utilization level must be limited below $1/(H^*-1)$. (We will refer to this bound as the *maximum allowable network utilization bound.*) For example, with $H^* = 3$ (a "small" network), the maximum network utilization must be kept below 50% of all link capacities; with $H^* = 11$ (a relatively "large" network), the maximum network utilization must be kept below 10% of all link capacities. Furthermore, as the network utilization level gets closer to $1/(H^*-1)$, the worst-case edge-to-edge delay bound approaches infinity.

IV. 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 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. 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 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. Therefore, 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. In the following we will analyze SETF(0) first, since its analysis is easier to present and follow. The general SETF(Γ) will be studied afterwards in Section IV-B.

A. SETF with Finest Time Granularity: SETF(0)

In this section we first establish performance bounds for SETF(0) and then discuss the packet state encoding issue.

A.1 Network Utilization Level and Edge-to-Edge Delay Bounds

We follow the same approach to establish performance bounds for a network of SETF(0) schedulers, as is employed for a network of FIFO schedulers in Section III.

Consider an arbitrary SETF(0) scheduler S of capacity C. As in Section III, let a_S^p and f_S^p denote, respectively, the time packet p arrives at and departs from S, and $Q(a_S^p)$ denote the amount of traffic serviced by the scheduler S between $[a_S^p, f_S^p]$. Note that unlike a FIFO scheduler, $Q(a_S^p)$ may *not* be equal to the amount of traffic queued at S at the arrival time of packet p. This is because a packet p' in the queue of scheduler S at the time of packet p arrival may have a time stamp $\omega_0^{p'} > \omega_0^p$. In addition, a packet p' arriving at S later than packet p (but before f_S^p) may have a time stamp $\omega_0^{p'} < \omega_0^p$, thus beginning service before packet p. Nonetheless, we can apply a similar argument as used in Lemma 1 to establish the following bound on $Q(a_S^p)$.

Lemma 4: For an SETF(0) scheduler S of capacity C, we have

$$Q(a_S^p) \leq \alpha C\{\tau^* - (a_S^p - a_1^p)\} + \beta C + L^{max}. \quad (11)$$

Comparing Lemma 4 with Lemma 1, we see that the upper bound on $Q(a_S^p)$ for an SETF(0) scheduler is reduced by $\alpha C(a_S^p - a_1^p)$ amount from that for an FIFO scheduler. This is not surprising, since any packet that is released into the network after $a_1^p = \omega_0^p$ will not take any service away from packet p at an SETF(0) scheduler (see Figure 4).

Lemma 5: Consider a packet p which traverses a path with h hops. Then for $i = 1, 2, \ldots, h$, we have,

$$f_i^p - a_1^p \leq \tau^*\{1-(1-\alpha)^i\} + \alpha^{-1}(\beta+\Delta)\{1-(1-\alpha)^i\}. \quad (12)$$

Using Lemma 5, we can establish the following main results for a network of SETF(0) schedulers.

[1] The proof of this theorem and the proofs of other results in the remainder of this paper can be found in [11].

Theorem 6: Consider a network of SETF(0) schedulers with a network diameter H^*. For $0 < \alpha < 1$, we have $\tau^* \leq \frac{\alpha^{-1}(\beta+\Delta)\{1-(1-\alpha)^{H^*-1}\}}{(1-\alpha)^{H^*-1}}$. Moreover, the worst-case edge-to-edge delay D^* is bounded above by,

$$D^* \leq \frac{\alpha^{-1}(\beta+\Delta)\{1-(1-\alpha)^{H^*}\}}{(1-\alpha)^{H^*-1}}. \quad (13)$$

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 desired: *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 $\alpha \to 1$. The worst-case edge-to-edge bounds for a FIFO network and an SETF(0) network (with $H^* = 8$) are shown (among other bounds) in Figure 5 as a function of the network utilization level α. In this example we assume that the capacity of all links is $10\,Gb/s$, and all packets have the same size $L = 1000$ bytes. We set the *network burstiness factor* β in a similar manner as in [3]: 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 $\beta_0 = 25\,ms$. From Figure 5, it is clear that for a given 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.

A.2 Time Stamp Encoding and Performance Trade-offs

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. 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[2]. We now investigate the problem of how many bits are needed to encode the packet time stamps.

Suppose that m bits are sufficient to encode the packet time stamps precisely. Then the time-stamp bit string wraps around every $2^m\iota$ units of time. Given that the worst-case edge-to-edge delay of a packet in the network is bounded above by D^*, we must have $2D^* \leq 2^m\iota$ so as to enable any SETF(0) scheduler to correctly distinguish and compare the time stamps of two different packets (see [11] for more discussions on this). From Theorem 6, we have

$$m \geq \log_2\{\frac{\alpha^{-1}(\beta+\Delta)\{1-(1-\alpha)^{H^*}\}}{((1-\alpha)^{H^*-1})\iota}\} + 1. \quad (14)$$

From (14), we see that 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.

[2] 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.

B. SETF with Coarser Time Granularity: SETF(Γ)

In this section we analyze the SETF(Γ) packet scheduling algorithm with coarser time granularity, i.e., $\Gamma > 0$, and illustrate how the time granularity affects the performance trade-offs of an SETF network. In particular, we demonstrate that using a coarser time granularity can potentially reduce the number of bits needed to encode the packet time stamps, albeit at the expenses of sacrificing the maximum allowable network utilization.

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, the 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. In the following we will employ the same approach as before to derive 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 minimum 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:

$$\min_{\text{all } p\text{'s}}\{a^p_{h^*+1} - a^p_1\} \geq \Gamma. \quad (15)$$

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. Hence, without loss of generality, in the rest of this section we assume that $1 \leq h^* < H^*-1$. Given this definition of h^*, we have the following bound on $Q(a^p_S)$, where the notations used in the lemma are defined as before:

Lemma 7: Consider an SETF(Γ) scheduler S with capacity C. Suppose S is the ith hop on the path of a packet p. Then

$$Q(a^p_S) \leq \alpha C\tau^* + \beta C, \text{ if } 1 \leq i \leq h^*, \quad (16)$$

and if $h^* < i \leq h$

$$Q(a^p_S) \leq \alpha C\{\tau^* - (a^p_S - a^p_{h^*+1})\} + \beta C + L^{max}, \quad (17)$$

where $a^p_{h^*+1}$ is the time packet p reaches its (h^*+1)th hop on its path.

Lemma 8: Consider a packet p which traverses a path with h hops. Then for $i = 1, 2, \ldots, h^*$,

$$f^p_i - a^p_1 \leq i(\alpha\tau^* + \beta); \quad (18)$$

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

$$\begin{aligned} f^p_i - a^p_1 \quad \leq \quad & h^*(\alpha\tau^* + \beta) + \tau^*\{1-(1-\alpha)^{i-h^*}\} \\ & + \alpha^{-1}(\beta+\Delta)\{1-(1-\alpha)^{i-h^*}\}. \quad (19) \end{aligned}$$

Applying Lemma 8, we obtain the following performance bounds for a network of SETF(Γ) schedulers.

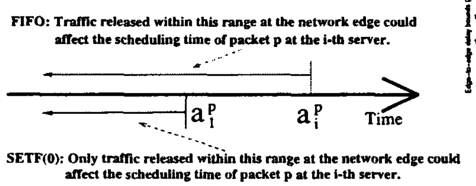

Fig. 4. Illustration of the different behaviors of FIFO and SETF(0).

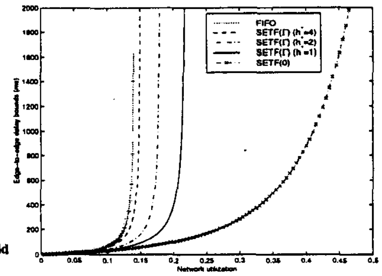

Fig. 5. Performance comparison: SETF vs. FIFO.

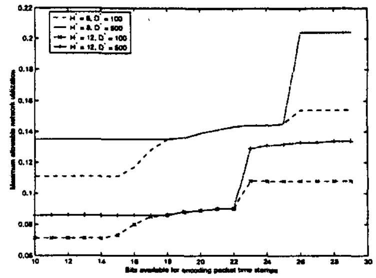

Fig. 6. No. of bits for encoding, network diameter, and maximum allowable network utilization.

Theorem 9: Consider a network of SETF(Γ) schedulers with a network diameter H^*. If the network utilization level α satisfies the following condition,

$$(1-\alpha)^{H^*-h^*-1} > \alpha h^*, \tag{20}$$

then

$$\tau^* \leq \frac{\beta h^* + \alpha^{-1}(\beta+\Delta)\{1-(1-\alpha)^{H^*-h^*-1}\}}{(1-\alpha)^{H^*-h^*-1} - \alpha h^*}. \tag{21}$$

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

$$D^* \leq \frac{\beta h^* + \alpha^{-1}(\beta+\Delta)\{1-(1-\alpha)^{H^*-h^*}\}}{(1-\alpha)^{H^*-h^*-1} - \alpha h^*}. \tag{22}$$

Note first that in Theorem 9, setting $h^* = 0$ yields the results for a network of SETF(0) schedulers, whereas setting $h^* = H^* - 1$ yields 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 accout for the non-preemptive property of an SETF scheduler). Hence Theorem 6 and Theorem 3 can be considered as two special cases of Theorem 9. In general, Theorem 9 states that with a coarser time granularity $\Gamma > 0$ (which determines h^*), we can no longer 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.

B.1 Time Stamp Encoding and Performance Trade-offs

In this section we show that using coarser time granularity can potentially reduce the number of bits needed for packet time stamp encoding. We also illustrate through numerical examples how time granularity affects the performance trade-offs of SETF(Γ) networks.

We first consider the problem of packet time stamp encoding. Using the same argument as in Section IV-A.2, for a given time granularity Γ and network utilization level α, the number of bits m needed for packet time stamp encoding must satisfy the following condition:

$$m \geq \log_2\{\frac{\beta h^* + \alpha^{-1}(\beta+\Delta)\{1-(1-\alpha)^{H^*-h^*}\}}{((1-\alpha)^{H^*-h^*-1} - \alpha h^*)\Gamma}\} + 1. \tag{23}$$

From Theorem 9, (23) and the definition of h^* (15), it is not too hard to 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. In all these studies, except for the network diameter H^* all other system parameters (link capacity, packet size, β) are the same as specified in Section IV-A.1.

Figure 5 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.

Next, 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 6, where networks with a combination of the network diameters $H^* = 8$ and $H^* = 12$ and delay bounds $D^* = 100\,ms$ and $D^* = 500\,ms$ 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 $D^* = 100\,ms$), the packet time stamp does no 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.

V. 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 as shown earlier. 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 7). 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*:

$$\omega_k^p := \omega_{k-1}^p + d^*, \qquad k = 1, \ldots, \kappa - 1, \tag{24}$$

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:* for $k = 1, \ldots, \kappa - 1$,

$$f_{kh^*}^p \leq \omega_k^p, \text{ and } f_h^p \leq \omega_\kappa^p := \omega_{\kappa-1}^p + d^*. \tag{25}$$

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 (25) 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 (25) and the packet time stamp update rule (24) 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: $[(n-1)\Gamma, n\Gamma), n = 1, 2, \ldots$, 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. 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 $k = 1, 2, \ldots, \kappa,$, 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 10: Consider an arbitrary scheduler S with capacity C in a network of DETF(Γ, h^*) schedulers. For any time interval $[\tau, t]$, let $\tilde{A}(\tau, t)$ be defined as above. Then

$$\tilde{A}(\tau, t) \leq \beta C + \alpha C(t - \tau + \Gamma). \tag{26}$$

Note that if $\Gamma = 0$, the bound on $\tilde{A}(\tau, t)$ is exactly the same as the edge traffic provisioning condition (2). Intuitively, (26) 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 $\alpha C \Gamma$ amount of traffic may (virtually) arrive at S, as opposed to (2) at the network edge.

From Lemma 10, 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 two special but *representative* cases—a network of DETF(0,1) schedulers and a network of DETF(Γ, 1). For the networks of DETF(Γ, h^*), $\Gamma > 0, h^* > 1$, see [11].

Theorem 11 (A Network of DETF(0,1) Schedulers) Consider a network of DETF(0,1) schedulers[3] with a network di-

[3]Note that a DETF(0,1) scheduler is a special case of the Virtual-Time Earliest-Deadline-First (VT-EDF) packet scheduling algorithm proposed in [10] under the *virtual time reference system* framework, where the delay parameter for all flows is set to d^*. In general, regarding the per-hop

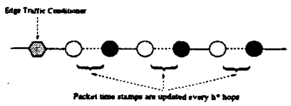

Fig. 7. Updating packet time stamps inside the network core.

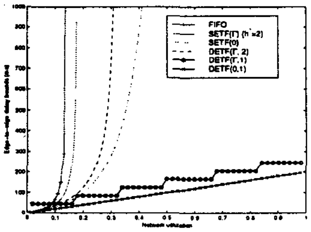

Fig. 8. Edge-to-edge delay bound comparison ($H^* = 8$).

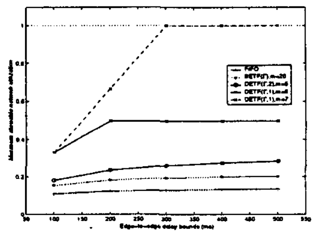

Fig. 9. Provisioning power of FIFO, SETF(Γ), DETF($\Gamma, 1$), and DETF($\Gamma, 2$) networks ($H^* = 8$).

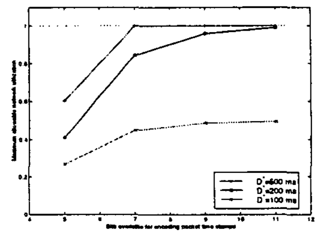

Fig. 10. Design and performance trade-offs for DETF($\Gamma, 1$) networks ($H^* = 8$).

ameter H^*. Let $d^* = \beta + \Delta$, then the reality condition (25) holds. Furthermore, for any $0 < \alpha < 1$, the worst-case edge-to-edge delay D^* is bounded above by $D^* \leq H^* d^* = H^*(\beta + \Delta)$.

Theorem 12 (A Network of DETF($\Gamma, 1$) Schedulers) Consider a network of DETF($\Gamma, 1$) schedulers with a network diameter H^*, where $\Gamma > 0$. Let $d^* = \lceil (\alpha\Gamma + \beta + \Delta)/\Gamma \rceil \Gamma$, then the reality condition (25) holds. Furthermore, for any $0 < \alpha < 1$, the worst-case edge-to-edge delay D^* is bounded above by $D^* \leq H^* d^* + \Gamma$.

From Theorem 11 and Theorem 12, 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 $d^* = \beta + \Delta$. With a coarser time granularity $\Gamma > 0$, *per-hop delay* is bounded by $d^* = \lceil (\alpha\Gamma + \beta + \Delta)/\Gamma \rceil \Gamma$, where the network utilization level determines the "additional delay" ($\alpha\Gamma$) that a packet may experience at each hop.

B. Packet State Encoding

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 11, we see that the number of bits m that is needed to encode the (dynamic) packet time stamps precisely must satisfy the following condition:

$$\begin{aligned} 2^{m-1}\iota &\geq H^*(\beta + \Delta), \text{ or} \\ m &\geq \log_2 H^* + \log_2[(\beta + \Delta)/\iota] + 1. \end{aligned}$$

Now consider a network of DETF($\Gamma, 1$) with a coarser time granularity $\Gamma > 0$. From Theorem 12, 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:

$$\begin{aligned} 2^{m-1}\Gamma &\geq H^* \lceil \frac{\alpha\Gamma + \beta + \Delta}{\Gamma} \rceil \Gamma + \Gamma, \text{ or} \\ m &\geq \log_2\{H^* \lceil \frac{\alpha\Gamma + \beta + \Delta}{\Gamma} \rceil + 1\} + 1. \quad (27) \end{aligned}$$

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 (27), 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., $\Gamma \in [0, (\beta + \Delta)/(1 - \alpha)]$. Any coarser time granularity $\Gamma > (\beta + \Delta)/(1 - \alpha)$ will not reduce the minimum number of bits, $\log_2\{H^* + 1\} + 1$, needed for packet time stamp encoding.

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 scheduling algorithms in support of guaranteed delay service. Lastly, we briefly touch on the issue of complexity/cost in implementing the aggregate scheduling algorithms. The network setting for all the studies is the same as before. 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 8, 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., $h^* = 2$). 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. 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.

scheduling behavior, DETF is close to a special case of SCED+ by Cruz [5]. However, SCED+ only considers discrete time and does not study the effect of number of bits available for packet state encoding on the performance of a network.

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(Γ, 2) network is 5. For the DETF(Γ, 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 9 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(Γ, 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(Γ, 1) networks only. In this study, we investigate the design and performance trade-offs in employing DETF(Γ, 1) networks to support guaranteed delay service. In Figure 10 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. From these results we see that with relatively a 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 bits.

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 [8]) with a number of FIFO queues. This particularly favors the DETF(Γ, 1) packet scheduling algorithms, since the number of bits needed for time stamp encoding can be kept small. However, compared to SETF, DETF(Γ, 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 [9]. 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.

VI. Conclusions

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.

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

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

[5] R. Cruz. Sced+: Efficient management of quality of service guarantees. In *Proc. IEEE INFOCOM*, San Francesco, CA, March 1998.

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

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

[8] 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.

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

[10] Z.-L. Zhang, Z. Duan, and Y. T. Hou. Virtual time reference system: A unifying scheduling framework for scalable support of guaranteed services. *IEEE Journal on Selected Areas in Communication*, Special Issue on Internet QoS, December 2000.

[11] 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.