Latency-Rate Servers: A General Model for Analysis of Traffic Scheduling Algorithms

Dimitrios Stiliadis and Anujan Varma, *Member, IEEE*

***Abstract*— In this paper, we develop a general model, called *Latency-Rate servers* ($\mathcal{LR}$ servers), for the analysis of traffic scheduling algorithms in broadband packet networks. The behavior of an $\mathcal{LR}$ server is determined by two parameters—the *latency* and the allocated *rate*. Several well-known scheduling algorithms, such as Weighted Fair Queueing, VirtualClock, Self-Clocked Fair Queueing, Weighted Round Robin, and Deficit Round Robin, belong to the class of $\mathcal{LR}$ servers. We derive tight upper bounds on the end-to-end delay, internal burstiness, and buffer requirements of individual sessions in an arbitrary network of $\mathcal{LR}$ servers in terms of the latencies of the individual schedulers in the network, when the session traffic is shaped by a token bucket. The theory of $\mathcal{LR}$ servers enables computation of tight upper bounds on end-to-end delay and buffer requirements in a heterogeneous network, where individual servers may support different scheduling architectures and under different traffic models.**

***Index Terms*—Delay bounds, fair queueing algorithms, performance bounds, traffic scheduling.**

I. Introduction

PROVIDING quality of service (QoS) guarantees in a packet network requires the use of traffic scheduling algorithms in the switches (or routers). The function of a scheduling algorithm is to select, for each outgoing link of the switch, the packet to be transmitted next from the available packets belonging to the sessions sharing the output link.

FIFO scheduling, perhaps the simplest to implement, does not provide any isolation between individual sessions which is necessary to achieve deterministic bandwidth guarantees. Several service disciplines are known in the literature for bandwidth allocation and transmission scheduling in output-buffered switches [4], [8], [10], [11], [15], [16], [18], [20], [23], [24], [28]. Many of these algorithms are also capable of providing deterministic delay guarantees when the burstiness of the session traffic is bounded (for example, shaped by a token bucket).

Since future networks are unlikely to be homogeneous in the type of scheduling algorithms employed by the individual switches (routers), a general model for the analysis of scheduling algorithms will be a valuable tool in the design and analysis of such networks. In this paper, we develop such a model for studying the worst-case behavior of individual sessions in a network of schedulers where the schedulers in the network may employ a broad range of scheduling algorithms. This approach enables us to calculate tight bounds on the end-to-end delay of individual sessions and the buffer sizes needed to support them in an arbitrary network of schedulers.

Our basic approach consists of defining a general class of schedulers, called *Latency-Rate* servers, or simply $\mathcal{LR}$ servers. The theory of $\mathcal{LR}$ servers provides a means to describe the worst-case behavior of a broad range of scheduling algorithms in a simple and elegant manner. This theory is based on the concept of a *busy period* of a session, a period of time during which the average arrival rate of the session remains at or above its reserved rate ρ_i. For a scheduling algorithm to belong to the $\mathcal{LR}$ class, it is only required that the *average* rate of service offered by the scheduler to a busy session, over every interval starting at time Θ from the beginning of the busy period, is at least equal to its reserved rate. The parameter Θ is called the *latency* of the scheduler. All the work-conserving schedulers that provide bandwidth guarentees, including Weighed Fair Queueing (also known as Packet-level Generalized Processor Sharing, or PGPS), VirtualClock, SCFQ, Weighted Round Robin, and Deficit Round Robin, exhibit this property and can therefore be modeled as $\mathcal{LR}$ servers.

The behavior of an $\mathcal{LR}$ scheduler is determined by two parameters—the *latency* and the *allocated rate*. The latency of an $\mathcal{LR}$ server may be seen as the worst-case delay seen by the first packet of the busy period of a session. The latency of a particular scheduling algorithm may depend on its internal parameters, its transmission rate on the outgoing link, and the allocated rates of various sessions. However, we show that the maximum end-to-end delay experienced by a packet in a network of schedulers can be calculated from only the latencies of the individual schedulers on the path of the session, and the traffic parameters of the session that generated the packet. Since the maximum delay in a scheduler increases directly in proportion to its latency, the model brings out the significance of using low-latency schedulers to achieve low end-to-end delays. Likewise, upper bounds on the queue size and burstiness of individual sessions at any point within the network can be obtained directly from the latencies of the schedulers. We also show how the latency parameter can be computed for a given scheduling algorithm by deriving the latencies of several well-known schedulers.

Manuscript received November 20, 1995; revised July 11, 1996; approved by IEEE/ACM Transactions on Networking Editor G. Sasaki. This work was supported by the National Science Foundation Young Investigator Award MIP-9257103. This paper was presented in part at the IEEE INFOCOM '96, April 1996.

D. Stiliadis is with Bell Laboratories, Lucent Technologies, Holmdel, NJ 07030 USA.

A. Varma is with the Computer Engineering Department, University of California at Santa Cruz, Santa Cruz, CA 95064 USA (e-mail: varma@cse.ucsc.edu).

Publisher Item Identifier S 1063-6692(98)08403-9.

The rest of this paper is organized as follows. In Section II, we define the class of $(\mathcal{LR})$ servers and analyze the worst-case service offered by a network of $\mathcal{LR}$ servers to a given session. We show that the worst-case offered service is independent of the input traffic characterization and only depends on the allocated rate and the internal parameters of the scheduler. In Section III, we derive upper bounds on the end-to-end delays, buffer requirements, and traffic burstiness of individual sessions in an arbitrary-topology network of $\mathcal{LR}$ servers when the session traffic is shaped by schemes such as a token bucket or a dual leaky bucket. In Section IV, we prove that several well-known scheduling algorithms, such as VirtualClock, Packet-level Generalized Processor Sharing (PGPS), Self-Clocked Fair Queueing (SCFQ), Weighted Round Robin (WRR), and Deficit Round Robin (DRR) all belong to the class of $\mathcal{LR}$ servers and derive their latencies. In Section V, we derive a slightly improved upper bound on end-to-end delay in a network of $\mathcal{LR}$ servers by bounding the service of the last server on the path more tightly. A comparison of the $\mathcal{LR}$ model with other approaches is presented in Section VI. Finally, Section VII contains some conclusions and directions for future work.

II. $\mathcal{LR}$ Servers

A. Definitions and Notations

We assume a packet switch where a set of V sessions share a common output link. We denote with ρ_i the rate allocated to session i. Servers are noncut-through devices. Let $A_i(\tau, t)$ denote the arrivals from session i during the interval $(\tau, t]$ and $W_i(\tau, t)$ the amount of service received by session i during the same interval. In a system based on the fluid model, both $A_i(\tau, t)$ and $W_i(\tau, t)$ are continuous functions of t. However, in the packet-by-packet model, we assume that $A_i(\tau, t)$ increases only when the last bit of a packet is received by the server; likewise, $W_i(\tau, t)$ is increased only when the last bit of the packet in service leaves the server. Thus, the fluid model may be viewed as a special case of the packet-by-packet model with infinitesimally small packets.

Definition 1: A **system busy period** is a maximal interval of time during which the server is never idle.

During a system busy period, the server is always transmitting traffic. Let $Q_i(t)$ represent the amount of session i traffic queued in the server at time t, that is,

$$Q_i(t) = A_i(0, t) - W_i(0, t).$$

A session is backlogged at time t if $Q_i(t) > 0$. That is,

Definition 2: A **backlogged period for session** i is any period of time during which traffic belonging to that session is continuously queued in the system.

Definition 3: A **session** i **busy period** is a maximal interval of time $(\tau_1, \tau_2]$ such that at any time $t \in (\tau_1, \tau_2]$ the accumulated arrivals of session i since the beginning of the interval do not fall below the total service received during the interval at a rate of exactly ρ_i. That is,

$$A_i(\tau_1, t) \geq \rho_i(t - \tau_1).$$

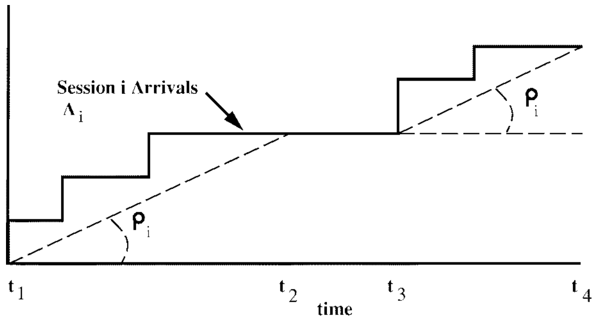

Fig. 1. Intervals $(t_1, t_2]$ and $(t_3, t_4]$ are two different busy periods of session i.

The session busy period is defined only in terms of the arrival function and the allocated rate, as illustrated in Fig. 1. It is important to realize the basic distinction between a session backlogged period and a session busy period. The latter is defined with respect to a hypothetical system where a backlogged session i is serviced at a constant rate ρ_i, while the former is based on the actual system where the instantaneous service can vary according to the state of the system. Thus, a busy period may contain intervals during which the actual backlog of session i traffic in the system is zero and the session is not receiving service.

Note that, when the same traffic distribution is applied to two different schedulers with identical reservations, the resulting backlogged periods can be quite different. This makes it difficult to use the session-backlogged period for analyzing a broad range of schedulers. The session busy period, on the other hand, depends only on the arrival pattern of the session and its allocated rate and can therefore be used as an invariant in the analysis of different schedulers. This is the fundamental reason why the following definition of an $\mathcal{LR}$ server is based on the service received by a session over a busy period. Since we are interested in a worst-case analysis of the system, the session busy period provides us a convenient means to bound the delay within the system.

We can now define the general class of $\mathcal{LR}$ servers. A server in this class is characterized by two parameters for each session i it services: *latency* Θ_i and *allocated rate* ρ_i. Let us assume that the jth busy period of session i starts at time τ. We denote by $W_{i,j}^{\mathcal{S}}(\tau, t)$ the total service provided by the server $\mathcal{S}$ to the traffic of the session that arrived during the jth busy period until time t. Note that the total service offered to session i in this interval $(\tau, t]$ may actually be more than $W_{i,j}^{\mathcal{S}}(\tau, t)$ since some traffic from a previous busy period, still queued in the system, may be serviced as well.

Definition 4: Let τ be the starting time of the jth busy period of session i in server $\mathcal{S}$ and τ^* the time at which the last bit of traffic arrived during the jth busy period leaves the server. Then, server $\mathcal{S}$ is an $\mathcal{LR}$ server if and only if a nonnegative constant $C_i^{\mathcal{S}}$ can be found such that, at every instant t in the interval $(\tau, \tau^*]$,

$$W_{i,j}^{\mathcal{S}}(\tau, t) \geq \max(0, \rho_i(t - \tau - C_i^{\mathcal{S}})).$$

The minimum nonnegative constant $C_i^{\mathcal{S}}$ satisfying the above inequality is defined as the *latency* of the server, denoted by $\Theta_i^{\mathcal{S}}$.

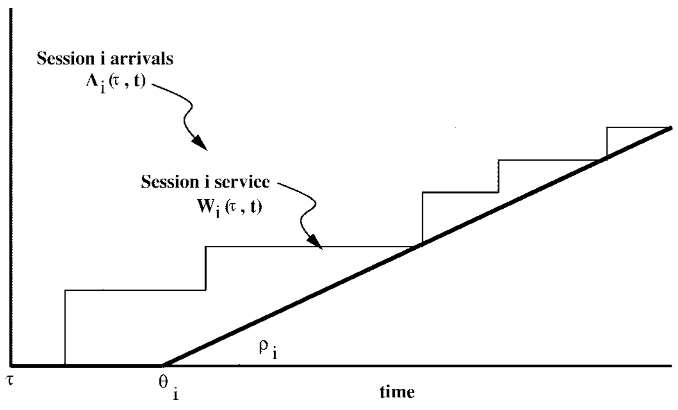

Fig. 2. An example of the behavior of an $\mathcal{LR}$ server.

The definition of $\mathcal{LR}$ servers involves only the two parameters, latency, and the allocated rate. The right-hand side of the above equation defines an envelope to bound the minimum service offered to session i during a busy period (Fig. 2). Notice that the restriction imposed is that the service provided to a session from the beginning of its busy period is lower bounded by this envelope. Thus, for a scheduling algorithm to belong to the $\mathcal{LR}$ class, it is only required that the *average* rate of service offered by the scheduler to a busy session, over every interval starting at time Θ_i^S from the beginning of the busy period, is at least equal to its reserved rate. This is much less restrictive than Generalized Processor Sharing (GPS), where the *instantaneous* bandwidth offered to a session is bounded [18]. That is, the lower bound on the service rate of GPS multiplexing holds for any interval $(\tau, t]$ that a session is backlogged, whereas in $\mathcal{LR}$ servers the restriction holds only for intervals starting at the beginning of the busy period. Therefore, GPS multiplexing is a special case of an $\mathcal{LR}$ server.

The latency parameter, in general, depends on the scheduling algorithm used as well as the allocated rate and traffic parameters of the session being served. For a particular scheduling algorithm, several parameters such as its transmission rate on the outgoing link, number of sessions sharing the link, and their allocated rates, may influence the latency. However, we will show that the maximum end-to-end delay experienced by a packet in a network of schedulers can be calculated from only the latencies of the individual schedulers on the path of the session and the traffic parameters of the session that generated the packet.

In our definition of $\mathcal{LR}$ servers, we made no assumptions on whether the server is based on the fluid model or the packet-by-packet model. The only requirement that we impose is that a packet is not considered as departing the server until its last bit has left the server. Therefore, packet departures must be considered as impulses. This assumption is needed to bound the arrivals into the next switch in a chain of schedulers. We will remove this assumption later from the last server of a chain to provide a slightly tighter bound on the end-to-end session delay in a network of schedulers. In a fluid system, we require that all schedulers operate on a fluid basis.

In the next section, we analyze the worst-case service offered to a session by a network of $\mathcal{LR}$ servers. The main result is that a network of servers can be analyzed as a single

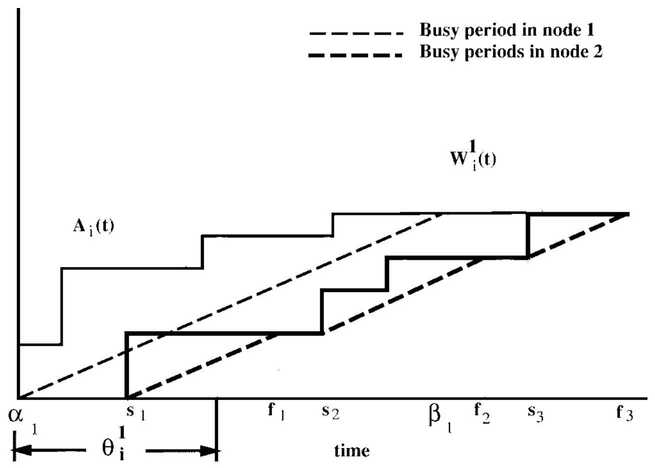

Fig. 3. Illustration of busy periods of a session in two servers in series. The busy period for session i in the first server is split into multiple busy periods in the second server. The busy period in the first server is $(\alpha_1, \beta_1]$. The packets arriving at the second server from this busy period form multiple busy periods $(s_1, f_1]$, $(s_2, f_2]$, and $(s_3, f_3]$ in the second server. The line with slope ρ_i that starts at Θ_i^1 bounds all these busy periods.

server, with rate equal to the lowest among the allocated rates of the servers in the path of the session, and latency equal to the sum of their latencies. This approach will enable us to provide bounds on delay and buffer requirements for a network of servers by simply analyzing an equivalent single $\mathcal{LR}$ server.

B. Analysis of a Network of $\mathcal{LR}$ Servers

The only restrictions that we impose on the network is that all the servers belong to the $\mathcal{LR}$ class. We assume that a minimum bandwidth of ρ_i is reserved for the session i at every node in its path. To derive tighter bounds for session delay in a network of $\mathcal{LR}$ servers, we first show that the worst-case offered service to a session in a network of two $\mathcal{LR}$ servers in series is the same as that in a single $\mathcal{LR}$ server whose latency is the sum of the latencies of the two servers it replaces. This result allows an arbitrary number of $\mathcal{LR}$ servers in the path of a session to be modeled by combining their individual latencies.

Analyzing two $\mathcal{LR}$ servers in series introduces a difficulty: if the first server has nonzero latency, the busy period of a session in the second server may not coincide with the corresponding busy period in the first server. That is, a packet that started a busy period in the first server may not start a busy period in the second, but instead might arrive while a busy period is in progress at the second server. Also, since the actual service rate seen by session i in the first server can be more than ρ_i, a single busy period in the first server may result in multiple busy periods in the second server. This is illustrated in Fig. 3. We will take these effects into account in our analysis of a network of $\mathcal{LR}$ servers.

In the following, we will use the term *network busy period* to mean a busy period of the session being analyzed in the *first* server on its path. Similarly, when we refer to *service offered by the network* to session i during an interval of time, we mean the amount of session-i traffic transmitted by the *last* server on the path during the interval under consideration. We will use $W_{i,j}(\tau, t)$ to denote the amount of service offered by the network to session i during an interval $(\tau, t]$ within its jth network busy period. Also, we will use $W^1_{i,j}(\tau_1, t_1)$ to denote the amount of service offered by the first server during an interval $(\tau_1, t_1]$ within its local busy period, and $W^2_{i,j}(\tau_2, t_2)$ the same parameter for the second server during an interval $(\tau_2, t_2]$ within its busy period.

We first prove the following lemma to bound the service provided by the network to a session during an interval within a network busy period.

Lemma 1: Consider a network of two $\mathcal{LR}$ servers S_1 and S_2 in series, with latencies $\Theta_i^{(\mathcal{S}_1)}$ and $\Theta_i^{(\mathcal{S}_2)}$, respectively. If ρ_i is the minimum rate allocated to session i in the network, the service offered by the network to the traffic of the jth network busy period of that session during an interval $(\tau, t]$ within the busy period is bounded as

$$W_{i,j}(\tau, t) \geq \max(0, \rho_i(t - \tau - (\Theta_i^{(\mathcal{S}_1)} + \Theta_i^{(\mathcal{S}_2)}))).$$

Proof: We will prove the lemma by induction on the number of network busy periods. Let us denote with $(\alpha_j, \beta_j]$ the starting and ending times of the jth network busy period of session i. $W^1_{i,j}(\alpha_j, t)$ denotes the service offered by the first server S_1 to the packets of the jth network busy period until time t. Note that the jth network busy period is the same as the jth-session busy period on the first server. However, the busy periods seen by the second server S_2 may be different from these periods. Let $(s_k, f_k]$ denote the kth busy period of session i in S_2. Then $W^2_{i,k}(s_k, t)$ is the service offered by S_2 during its kth busy period until time t.

Let τ_k denote the time at which packets from the kth busy period of session i in S_2 start service in S_2. Similarly, let t_k be the instant at which the last packet from the kth busy period in S_2 is completely serviced by S_2. Then, it is easy to observe that $\tau_k \geq s_k$ and $t_k \leq \tau_{k+1}$.

Base step: Consider any time t during the first network busy period, that is, $\alpha_1 \leq t \leq \beta_1$. When the first busy period for session i starts, there is no other traffic from that session in the system. As observed earlier, the first network busy period of session i may result in multiple busy periods in the second server. Let $(s_1, f_1]$, $(s_2, f_2], \cdots, (s_m, f_m]$ be the successive busy periods in the second server in which packets from the first network busy period were serviced.

Let t^* denote the last instant of time at which a packet from the first network busy period was in service in S_2. We need to show that, for any time t, $\alpha_1 \leq t \leq t^*$

$$W_{i,1}(\alpha_1, t) \geq \max(0, \rho_i(t - \alpha_1 - (\Theta_i^{(\mathcal{S}_1)} + \Theta_i^{(\mathcal{S}_2)}))).$$

We need to consider three separate cases for t.

Case 1: $t \leq \tau_1$. In this case, no service has occurred in the second server, that is, $W_{i,1}(\alpha_1, t) = 0$. Also, $s_1 \leq \alpha_1 + \Theta_i^{(\mathcal{S}_1)}$, and $\tau_1 \leq s_1 + \Theta_i^{(\mathcal{S}_2)}$. Therefore,

$$t - \alpha_1 \leq \Theta_i^{(\mathcal{S}_1)} + \Theta_i^{(\mathcal{S}_2)}. \tag{2.1}$$

Hence, we can write

$$W_{i,1}(\alpha_1, t) = \max(0, \rho_i(t - \alpha_1 - (\Theta_i^{(\mathcal{S}_1)} + \Theta_i^{(\mathcal{S}_2)}))). \tag{2.2}$$

Case 2: $\tau_k \leq t \leq t_k$, $1 \leq k \leq m$ (within the kth busy period in S_2). Note that, during the last (mth) busy period in S_2, the traffic serviced by S_2 may include packets from the second network busy period. Therefore, for the mth local busy period in S_2, we consider only t in the range $\tau_m \leq t < t^*$, where t^* is the last instant of time when a packet from the first network busy period was serviced by S_2

$$W_{i,1}(\alpha_1, t) = W_{i,1}(\alpha_1, \tau_k) + W_{i,1}(\tau_k, t). \tag{2.3}$$

But

$$W_{i,1}(\alpha_1, \tau_k) = W^1_{i,1}(\alpha_1, s_k)$$

and

$$W_{i,1}(\tau_k, t) = W^2_{i,k}(\tau_k, t) = W^2_{i,k}(s_k, t).$$

Substituting in (2.3)

$$\begin{aligned} W_{i,1}(\alpha_1, t) &= W^1_{i,1}(\alpha_1, s_k) + W^2_{i,k}(s_k, t) \\ &\geq \max(0, \rho_i(s_k - \alpha_1 - \Theta_i^{(\mathcal{S}_1)}) \\ &\quad + \max(0, \rho_i(t - s_k - \Theta_i^{(\mathcal{S}_2)})) \\ &\geq \max(0, \rho_i(t - \alpha_1 - (\Theta_i^{(\mathcal{S}_1)} + \Theta_i^{(\mathcal{S}_2)}))). \end{aligned}$$

Case 3: $t_k \leq t \leq \tau_{k+1}$ (between busy periods in S_2). We can write

$$W_{i,1}(\alpha_1, t) = W_{i,1}(\alpha_1, \tau_{k+1}) - W_{i,1}(t, \tau_{k+1}). \tag{2.4}$$

Since no packet arrives at the second server between its busy periods, the second term on the right-hand side is zero. Also, $W_{i,1}(\alpha_1, \tau_{k+1}) = W^1_{i,1}(\alpha_1, s_{k+1})$. Therefore, (2.4) reduces to

$$\begin{aligned} W_{i,1}(\alpha_1, t) &= W^1_{i,1}(\alpha_1, s_{k+1}) \\ &\geq \max(0, \rho_i(s_{k+1} - \alpha_1 - \Theta_i^{(\mathcal{S}_1)})). \end{aligned}$$

But, $t \leq \tau_{k+1} \leq s_{k+1} + \Theta_i^{(\mathcal{S}_2)}$, or $s_{k+1} \geq t - \Theta_i^{(\mathcal{S}_2)}$. Therefore, we can rewrite (2.5) as

$$W_{i,1}(\alpha_1, t) \geq \max(0, \rho_i(t - \alpha_1 - (\Theta_i^{(\mathcal{S}_1)} + \Theta_i^{(\mathcal{S}_2)}))).$$

Inductive step: We assume that for all network busy periods $1, \cdots, n$, the lemma is true, and thus

$$W_{i,n}(\alpha_n, t) \geq \max(0, \rho_i(t - \alpha_n - (\Theta_i^{(\mathcal{S}_1)} + \Theta_i^{(\mathcal{S}_2)})))$$

for $\alpha_n \leq t \leq \beta_n$. We will now prove that, for the $(n+1)$th network busy period,

$$W_{i,n+1}(\alpha_{n+1}, t) \geq \max(0, \rho_i(t - \alpha_{n+1} - (\Theta_i^{(\mathcal{S}_1)} + \Theta_i^{(\mathcal{S}_2)})))$$

for $\alpha_{n+1} \leq t \leq \beta_{n+1}$. In the simple case, the first session-i packet of the $(n+1)$th network busy period also starts a busy period in the second server. This case can be handled exactly as in the base step. The more difficult case occurs if, when the $(n+1)$th network busy period starts, some packets of session i from earlier busy periods are still backlogged in the second server. That is, the beginning of a network busy period of session i may not always coincide with the beginning of a busy period for S_2, as was the case in the base step. However, we will now show that this does not affect our analysis.

Let us assume that the mth busy period of S_2 is in progress when the first packet from the $(n+1)$th network busy period arrives at S_2. Assume that the mth busy period of S_2 started at time s_m, and the first packet from the $(n+1)$th network busy period was serviced by S_2 at time τ. Then, at any time t during the $(n+1)$th network busy period

$$\begin{aligned} W_{i,n+1}(\alpha_{n+1}, t) &= W_{i,n+1}(\tau, t) \\ &= W^2_{i,m}(s_m, t) - W^2_{i,m}(s_m, \tau). \end{aligned} \tag{2.6}$$

Note that the mth busy period currently in progress in S_2 may contain packets from multiple network busy periods, since multiple busy periods of S_1 may merge into a single busy period in S_2. Assume that the mth busy period of S_2 contained packets from the network busy periods $n-L$, $n-L+1, \cdots, n+1$. The total number of packets that are served during the mth busy period of the second server until time τ is equal to the total number of packets that arrived during the network busy periods $n-L$, $n-L+1, \cdots, n$, minus the packets that were serviced by the first server during the interval $(\alpha_{n-L}, s_m]$. Thus

$$\begin{aligned} W^2_{i,m}(s_m, \tau) &= \sum_{j=n-L}^{n} A_i(\alpha_j, \beta_j) - W^1_{i,n-L}(\alpha_{n-L}, s_m) \\ &\le \sum_{j=n-L}^{n} \rho_i(\beta_j - \alpha_j) \\ &\quad \cdot \max(0, \rho_i(s_m - \alpha_{n-L} - \Theta_i^{(\mathcal{S}_1)})) \\ &\le \rho_i(\beta_n - \alpha_{n-L}) \\ &\quad \cdot \max(0, \rho_i(s_m - \alpha_{n-L} - \Theta_i^{(\mathcal{S}_1)})) \\ &\le \rho_i(\beta_n - s_m + \Theta_i^{(\mathcal{S}_1)}). \end{aligned} \tag{2.7}$$

From (2.6) and (2.7), and the definition of $\mathcal{LR}$ servers,

$$\begin{aligned} W_{i,n+1}(\alpha_{n+1}, t) &= W^2_{i,m}(s_m, t) - W^2_{i,m}(s_m, \tau) \\ &\ge \max(0, \rho_i(t - s_m - \Theta_i^{(\mathcal{S}_2)}) \\ &\quad - \rho_i(\beta_n - s_m + \Theta_i^{(\mathcal{S}_1)})) \\ &= \max(0, \rho_i(t - \beta_n - (\Theta_i^{(\mathcal{S}_1)} + \Theta_i^{(\mathcal{S}_2)}))) \\ &\ge \max(0, \rho_i(t - \alpha_{n+1} - (\Theta_i^{(\mathcal{S}_1)} + \Theta_i^{(\mathcal{S}_2)})). \end{aligned}$$

The last inequality holds since $\beta_n \le \alpha_{n+1}$.

Now, if the $(n+1)$th network busy period is later split into multiple busy periods in S_2, we can use the same approach used in the base step for subsequent busy periods in S_2 to complete the proof. $\square$

Lemma 1 asserts that the service offered to a session in a network of two $\mathcal{LR}$ servers in series is no less in the worst case than the service offered to the session by a single $\mathcal{LR}$ server whose latency is equal to the sum of the latencies of the two servers it replaces. In the above proof, we assumed that each server in the path of the session allocates identical rates to the session. If this is not the case, we can simply use the *minimum* among the allocated rates to estimate the worst-case offered service.

Lemma 2: An $\mathcal{LR}$ server that can guarantee rate g_i with latency Θ_i can also guarantee rate $g'_i \le g_i$ with latency Θ_i.

Since the slope of the bounding envelope of the $\mathcal{LR}$ server depends on the rate, a change in the guaranteed rate from g_i to g'_i alters the busy periods of the session in the server. However, it is easy to see that an envelope with slope g'_i and x-intercept Θ_i is still a lower bound for the service offered to the session during a busy period based on the rate g'_i. A formal proof of this lemma can be found in [21]. Using lemmas 1 and 2, we can state the following main theorem.

Theorem 1: Let τ be the start of the jth network busy period of session i in a network of $\mathcal{LR}$ servers. If ρ_i is the minimum bandwidth allocated to session i in the network, the service offered to packets of the jth network busy period after the kth node in the network is given by

$$W_{i,j}^{S_k}(\tau, t) \ge \max(0, \rho_i \left(t - \tau - \sum_{j=1}^{k} \Theta_i^{(S_j)}) \right)$$

where $\Theta_i^{(S_j)}$ is the latency of the jth server in the network for session i.

Thus, we have established that the worst-case behavior of a network of $\mathcal{LR}$ servers can be analyzed simply by studying the behavior of a single equivalent $\mathcal{LR}$ server. Note that the results so far did not make any assumptions on input traffic. Thus, the $\mathcal{LR}$ model can be used to derive upper bounds on delay and buffer requirements for any traffic model. We illustrate this in the next section by deriving upper bounds on delays and buffer requirements for two well-known traffic models.

III. Delay and Backlog Analysis of Networks of $\mathcal{LR}$ Servers

In this section, we will demonstrate the utility of the model of $\mathcal{LR}$ servers by deriving bounds on end-to-end delays and buffer requirements for input traffic defined by two common shaping schemes—token bucket and dual leaky bucket.

A. Token-Bucket-Shaped Traffic

Since we have already shown that an arbitrary network of $\mathcal{LR}$ servers can be analyzed by considering an equivalent single $\mathcal{LR}$ server, we will study only the behavior of a session in a single $\mathcal{LR}$ server. We will assume that the input traffic of the session we analyze is token-bucket smoothed and the allocated rate is at least equal to the average arrival rate. That is, if i is the session under observation, its arrivals at the input of the network during the interval $(\tau, t]$ satisfy the inequality

$$A_i(\tau, t) \le \sigma_i + \rho_i(t - \tau) \tag{3.8}$$

where σ_i and ρ_i denote its burstiness and average rate, respectively. We make no assumptions on the input traffic of other sessions.

Assume a set of V sessions sharing the same output link of an $\mathcal{LR}$ server. First, we show that the packets of a busy period in the server complete service no later than $\Theta_i^{\mathcal{S}}$ after the busy period ends.

Lemma 3: Let $(t_1, t_2]$ be a busy period of session-i in server $\mathcal{S}$. All packets that arrived from session i during the busy period will be serviced by time $t_2 + \Theta_i^{\mathcal{S}}$, where $\Theta_i^{\mathcal{S}}$ is the latency of the server.

Proof: Let us assume that the last packet of the jth busy period completes service at time t. Then, at t,

$$A_i(t_1, t_2) = W_{i,j}^{\mathcal{S}}(t_1, t). \tag{3.9}$$

We know from the definition of the busy period that

$$A_i(t_1, t_2) = \rho_i(t_2 - t_1). \tag{3.10}$$

From the definition of $\mathcal{LR}$ servers and (3.9) and (3.10),

$$\rho_i(t_2 - t_1) \geq \rho_i(t - t_1 - \Theta_i^{\mathcal{S}}) \tag{3.11}$$

or equivalently

$$t \leq t_2 + \Theta_i^{\mathcal{S}}. \tag{3.12}$$

□

The following theorem bounds the queueing delays within the server:

Theorem 2: If $\mathcal{S}$ is an $\mathcal{LR}$ server and the input traffic for session i is token-bucket-shaped, then the maximum delay $D_i^{\mathcal{S}}$ of any packet of session i in $\mathcal{S}$ is given by

$$D_i^{\mathcal{S}} \leq \frac{\sigma_i}{\rho_i} + \Theta_i^{\mathcal{S}}. \tag{3.13}$$

Let us assume that $D_i^{\mathcal{S}}$ is the delay obtained for a packet that arrived at time t^* during the jth busy period. This means that the packet was serviced at time $t^* + D_i^{\mathcal{S}}$. Hence, the amount of service offered to the session until time $t^* + D_i^{\mathcal{S}}$ is equal to the amount of traffic that arrived from the session until time t^*. That is, if s_j is the beginning of the jth busy period

$$W_{i,j}^{\mathcal{S}}(s_j, t^* + D_i^{\mathcal{S}}) = A_i(s_j, t^*). \tag{3.14}$$

From the definition of an $\mathcal{LR}$ server,

$$W_{i,j}^{\mathcal{S}}(s_j, t^* + D_i^{\mathcal{S}}) \geq \rho_i(t^* + D_i^{\mathcal{S}} - s_j - \Theta_i^{\mathcal{S}}). \tag{3.15}$$

From (3.14), (3.15), and the token-bucket constraint (3.8), we have

$$\rho_i(t^* + D_i^{\mathcal{S}} - s_j - \Theta_i^{\mathcal{S}}) \leq \sigma_i + \rho_i(t^* - s_j) \tag{3.16}$$

or equivalently

$$D_i^{\mathcal{S}} \leq \frac{\sigma_i}{\rho_i} + \Theta_i^{\mathcal{S}}. \tag{3.17}$$

□

Note that the delay bound obtained here is independent of whether the server is work-conserving. For the special case when all the servers are work-conserving, we can derive similar general bounds on the maximum amount of buffering that must be allocated to the session in each server to guarantee zero loss.

Theorem 3: If $\mathcal{S}$ is an $\mathcal{LR}$ server, the following bounds hold for a session i serviced by $\mathcal{S}$.

1) If $Q_i^{\mathcal{S}}(t)$ is the backlog of session i in $\mathcal{S}$ at time t, then

$$Q_i^{\mathcal{S}}(t) \leq \sigma_i + \rho_i \Theta_i^{\mathcal{S}}. \tag{3.18}$$

2) The output traffic conforms to the token-bucket model with parameters $(\sigma_i + \Theta_i^{\mathcal{S}} \rho_i, \rho_i)$.

Proof: We will first prove the upper bound on session backlog. Let $(s_j, f_j]$ denote the jth busy period of the server. Let us denote with $W_{i,j}^{\mathcal{S}}(\tau, t)$ the service offered to the packets of the jth busy period of session i in the interval $(\tau, t]$, with $\tau \geq s_j$ and $t \leq f_j + \Theta_i^{\mathcal{S}}$. We will denote by $W_i^{\mathcal{S}}(\tau, t)$ the total service offered to session i during the same interval of time. Note that $W_i^{\mathcal{S}}(\tau, t) \geq W_{i,j}^{\mathcal{S}}(\tau, t)$, since, during the interval $(\tau, t]$ the server may transmit packets of a previous busy period as well.

During a system busy period, let us denote with τ_j the time at which the last packet of the jth busy period completes service. At this point, all backlogged packets belong to busy periods that started after time f_j. Also, by lemma 3, $\tau_j \leq f_j + \Theta_i^{\mathcal{S}}$. We will prove the theorem by induction over the time intervals $(\tau_{j-1}, \tau_j]$.

Base step: Assume that the first session-i busy period started at time $s_1 = \tau_0$. Since this is the first busy period for session i, there are no other packets backlogged from session i at τ_0. By the definition of $\mathcal{LR}$ servers, we know that for every instant t with $\tau_0 \leq t \leq \tau_1$

$$W_{i,1}^{\mathcal{S}}(\tau_0, t) = W_{i,1}^{\mathcal{S}}(s_1, t) \geq \max(0, \rho_i(t - s_1 - \Theta_i^{\mathcal{S}})).$$

The backlog at time t is defined as the total amount of traffic that arrived since the beginning of the system busy period minus those packets that were serviced. Therefore,

$$Q_i(t) = A_i(s_1, t) - W_{i,1}^{\mathcal{S}}(s_1, t). \tag{3.19}$$

From (3.18), (3.19), and the definition of an $\mathcal{LR}$ server,

$$\begin{aligned} Q_i^{\mathcal{S}}(t) &\leq \sigma_i + \rho_i(t - s_1) - \max(0, \rho_i(t - s_1 - \Theta_i^{\mathcal{S}})) \\ &\leq \sigma_i + \rho_i \Theta_i^{\mathcal{S}}. \end{aligned} \tag{3.20}$$

Inductive step: We will assume that the theorem holds until the end of the nth busy period. That is, at any instant $t \leq \tau_n$, $Q_i^{\mathcal{S}}(t) \leq \sigma_i + \rho_i \Theta_i^{\mathcal{S}}$. We will now prove that it holds also during the interval $(\tau_n, \tau_{n+1}]$. From the definition of τ_n and τ_{n+1}, only packets from the $(n+1)$th busy period would be serviced after time τ_n, and by time τ_{n+1} all the packets that belong in the $(n+1)$th busy period would have completed service. In addition, no packet from the $(n+1)$th busy period has been serviced before time τ_n. Therefore, for any instant t with $\tau_n < t \leq \tau_{n+1}$, we can write

$$W_{i,n+1}^{\mathcal{S}}(\tau_n, t) = W_{i,n+1}^{\mathcal{S}}(s_{n+1}, t).$$

The amount of packets that are backlogged in the server is equal to those packets that arrived after the end of the nth busy period minus those packets that were serviced after time τ_n, but we know that, since s_{n+1} is the beginning of the $(n+1)$th busy period, the first packet of the busy period arrived at time s_{n+1}. Therefore,

$$\begin{aligned} Q_i^{\mathcal{S}}(t) &\leq A_i(s_{n+1}, t) - W_{i,n+1}^{\mathcal{S}}(\tau_n, t) \\ &\leq \sigma_i + \rho_i(t - s_{n+1}) - \max(0, \rho_i(t - s_{n+1} - \Theta_i^{\mathcal{S}})) \\ &\leq \sigma_i + \rho_i \Theta_i^{\mathcal{S}}. \end{aligned}$$

Upper bound on burstiness of output traffic: We will now prove that the output traffic of the scheduler conforms to the token-bucket model as well. It is sufficient to show that, during any interval $(\tau, t]$,

$$W_i(\tau, t) \leq \sigma_i + \rho_i \Theta_i^{\mathcal{S}} + \rho_i(t - \tau).$$

Let us denote with σ_i^{out} the burstiness of session i at the output of the switch. We will try to find the maximum value for this burstiness. Let us denote with $c_i(\tau)$ the amount of tokens that remain at time τ in the token bucket shaping the incoming traffic. $Q_i^{\mathcal{S}}(\tau)$ is the amount of backlogged packets in the server at time τ. We assume that the input links have infinite capacity; therefore, at time $\tau+$ it is possible that an amount of $c_i(\tau)$ packets is added to the server queue and no packet is serviced. Thus, the maximum possible backlog at time τ is $c_i(\tau) + Q_i^{\mathcal{S}}(\tau)$; but we already proved that the maximum backlog of the session i is bounded by $\sigma_i + \rho_i \Theta_i^{\mathcal{S}}$. Therefore,

$$c_i(\tau) + Q_i^{\mathcal{S}}(\tau) \leq \sigma_i + \rho_i \Theta_i^{\mathcal{S}}. \tag{3.21}$$

The arrivals $A_i(\tau, t)$ cannot exceed the amount of tokens in the token bucket at time τ plus the amount of tokens that arrived during the interval $(\tau, t]$, minus the amount of tokens remaining in the bucket at time t. That is,

$$A_i(\tau, t) \leq c_i(\tau) + \rho_i(t - \tau) - c_i(t). \tag{3.22}$$

We can also calculate the service offered to session i in the interval $(\tau, t]$ as

$$\begin{aligned} W_i^{\mathcal{S}}(\tau, t) &= Q_i(\tau) + A_i(\tau, t) - Q_i(t) \\ &< Q_i(\tau) + A_i(\tau, t). \end{aligned} \tag{3.23}$$

Then, from (3.22) and (3.23),

$$\begin{aligned} W_i^{\mathcal{S}}(\tau, t) &\leq Q_i(\tau) + c_i(\tau) + \rho_i(t - \tau) - c_i(t) \\ &\leq (\sigma_i + \rho_i \Theta_i^{\mathcal{S}}) + \rho_i(t - \tau). \end{aligned} \tag{3.24}$$

Therefore,

$$\sigma_i^{\text{out}} \leq \sigma_i + \rho_i \Theta_i^{\mathcal{S}}. \tag{3.25}$$

It is easy to show that this bound is tight when the outgoing link has infinite capacity. We will assume that the server is work-conserving. Hence, if session i is the only active session in the system, it will be serviced exclusively. Let us assume that the maximum backlog for this session is reached at time t and that session i is the only active session in the server at that time. The server will then service the backlogged packets instantaneously, resulting in an output burstiness of

$$\sigma_i^{\text{out}} \geq \sigma_i + \rho_i \Theta_i^{\mathcal{S}}. \tag{3.26}$$

From (3.25) and (3.26), we can conclude that $\sigma_i^{\text{out}} = \sigma_i + \rho_i \Theta_i^{\mathcal{S}}$. □

We can now easily extend the results obtained for one scheduler by using Theorems 1 and 3. We first prove the following lemma.

Lemma 4: The traffic process of a session after the kth node in a chain of $\mathcal{LR}$ servers conforms to the token-bucket model with parameters

$$\sigma_i + \rho_i \sum_{j=1}^{k} \Theta_i^{(S_j)} \quad \text{and} \quad \rho_i$$

where $\Theta_i^{(S_j)}$ is the latency of the jth scheduler in the path of the session.

Proof: We already proved in Theorem 3 that the output traffic of an $\mathcal{LR}$ server conforms to the token-bucket model with parameters $(\sigma_i + \rho_i \Theta_i^{\mathcal{S}}, \rho_i)$. This means that the input traffic at the next node conforms to the same model. Therefore, at the kth node, the input traffic of session i will conform to the token bucket model with burstiness $\sigma_i + \rho_i \sum_{j=1}^{k-1} \Theta_i^{(S_j)}$. Hence, by Theorem 3, the burstiness of the output traffic of node k will be bounded by

$$\sigma_i^{\text{out}} \leq \sigma_i + \rho_i \sum_{j=1}^{k} \Theta_i^{(S_j)}. \tag{3.27}$$

□

This is an important result that allows us to bound the burstiness of session-i traffic at each point in the network. Notice that the increase in burstiness that a session may experience in the network is proportional to the sum of the latencies of the servers it has traversed. Therefore, even a session with no burstiness at the input of the network may accumulate significant burstiness in the network because of the latency of the servers.

We can now state the following lemma that will bound the backlog of session i in each node of the network.

Lemma 5: The maximum backlog $Q_i^{(S_k)}(t)$ in the kth node of a session is bounded as

$$Q_i^{(S_k)}(t) \leq \sigma_i + \rho_i \sum_{j=1}^{k} \Theta_i^{(S_j)}.$$

Proof: Follows directly from Theorem 3 and Lemma 4.

As a result of the increased burstiness, the maximum backlog and therefore the maximum buffer requirements for each node in the network are also proportional to the combined latency of the servers. In Lemma 5, we bounded the maximum backlog for session i in any node of the network. This bound may be seen as the maximum number of buffers needed in the corresponding node of the network to guarantee zero packet loss for session i. However, this does not mean that we can bound the maximum number of packets in the network for session i by adding the backlog bounds of each switch along the path. Such a bound does not take into account the correlation among arrival processes at the individual nodes and therefore can be very loose.

Using Theorem 1, we can bound the end-to-end delays of session i if the input traffic is token-bucket-shaped and the average arrival rate is less than ρ_i.

Theorem 4: The maximum delay D_i of session i in a network of $\mathcal{LR}$ servers, consisting of k servers in series, is bounded as

$$D_i \leq \frac{\sigma_i}{\rho_i} + \sum_{j=1}^{k} \Theta_i^{(S_j)} \tag{3.28}$$

where $\Theta_i^{(S_j)}$ is the latency of the jth server in the network for session i.

Proof: From Theorem 1, we can treat the whole network as equivalent to a single $\mathcal{LR}$ server with latency equal to the sum of their latencies. Hence, the result follows directly from Theorem 3. □

This maximum delay is independent of the topology of the network. The bound is also much tighter than what could be obtained by analyzing each server in isolation. Note that the end-to-end delay bound is a function of only two parameters: the burstiness of the session traffic at the source and the latencies of the individual servers on the path of the session. Since we assumed only that each of the servers in the network belongs to the $\mathcal{LR}$ class, these results are more general than the delay bounds due to Parekh and Gallager [19]. In the next section, we will show that many well-known work-conserving schedulers are in fact $\mathcal{LR}$ servers. Thus, our delay bound applies to almost any network of schedulers.

The delay bound in (3.28) shows that there are two ways to minimize delays and buffer requirements in a network of $\mathcal{LR}$ servers: 1) allocate more bandwidth to a session, thus reducing the term σ_i/ρ_i or 2) use $\mathcal{LR}$ servers with low latencies. Since the latency is accumulated through multiple nodes, the second approach is preferred in a large network. The first approach reduces the utilization of the network, thus allowing only a smaller number of simultaneous sessions to be supported than would be possible with minimum-latency servers. Minimizing the latency also minimizes the buffer requirements of the session at the individual servers in the network.

Note that the latency of a server depends, in general, on its internal parameters and the bandwidth allocation of the session under consideration. In addition, the latency may also vary with the number of active sessions and their allocations. Such a dependence of the latency of one session on other sessions indicates the poor isolation properties of the scheduler. Likewise, in some schedulers, the latency may depend strongly on its internal parameters and less on the bandwidth allocation of the session under observation. Such schedulers do not allow control of the latency of a session by controlling its bandwidth allocation.

B. Dual Leaky-Bucket-Shaped Traffic

Since the definition of an $\mathcal{LR}$ server is not based on any assumptions on the input traffic, it is easy to derive delay bounds for traffic distributions other than the (σ, ρ) model. For example, when the peak rate of the source is known, a modified upper bound on the delay of an $\mathcal{LR}$ server can be obtained. Let us denote with g_i the service rate allocated to session i, and let ρ_i and P_i, respectively, denote the average and peak rate at the source of session i. The arrivals at the input of the server during the interval $(\tau, t]$ now satisfy the inequality

$$A_i(\tau, t) \leq \min(\sigma_i + \rho_i(t-\tau), P_i(t-\tau)). \tag{3.29}$$

We can prove the following lemma:

Lemma 6: The maximum delay D_i of a session i in an $\mathcal{LR}$ server, where the peak rate of the source is known, is bounded as

$$D_i^{\mathcal{S}} \leq \left(\frac{P_i - g_i}{P_i - \rho_i}\right)\left(\frac{\sigma_i}{g_i}\right) + \Theta_i^{\mathcal{S}}. \tag{3.30}$$

Proof: Let us assume that $D_i^{\mathcal{S}}$ is the delay experienced by a packet that arrived at time t^* during the jth busy period. This means that the packet was serviced at time $t^* + D_i^{\mathcal{S}}$. Hence, the amount of service offered to the session until time $t^* + D_i^{\mathcal{S}}$ is equal to the amount of traffic that arrived from the session until time t. If s_j denotes the beginning of the jth busy period

$$W_{i,j}^{\mathcal{S}}(s_j, t^* + D_i^{\mathcal{S}}) = A_i(s_j, t^*). \tag{3.31}$$

From (3.29), this becomes

$$W_{i,j}^{\mathcal{S}}(s_j, t^* + D_i^{\mathcal{S}}) \leq \min(\sigma_i + \rho_i(t^* - s_j), P_i(t^* - s_j)). \tag{3.32}$$

From the definition of the $\mathcal{LR}$ server

$$W_{i,j}^{\mathcal{S}}(s_j, t^* + D_i^{\mathcal{S}}) \geq g_i(t^* + D_i^{\mathcal{S}} - s_j - \Theta_i^{\mathcal{S}}). \tag{3.33}$$

From (3.32) and (3.33), we have

$$\begin{aligned} &g_i(t^* + D_i^{\mathcal{S}} - s_j - \Theta_i^{\mathcal{S}}) \\ &\qquad \leq \min(\sigma_i + \rho_i(t^* - s_j), P_i(t^* - s_j)). \end{aligned} \tag{3.34}$$

Case 1: When $P_i(t^* - s_j) \leq \sigma_i + \rho_i(t^* - s_j)$, we have

$$D_i^{\mathcal{S}} \leq \left(\frac{P_i - g_i}{g_i}\right)(t^* - s_j) + \Theta_i^{\mathcal{S}} \tag{3.35}$$

and

$$P_i(t^* - s_j) \leq \sigma_i + \rho_i(t^* - s_j) \quad \text{or} \quad t^* - s_j \leq \frac{\sigma_i}{P_i - \rho_i}.$$

Substituting for $(t^* - s_j)$ in (3.35), we get

$$D_i^{\mathcal{S}} \leq \left(\frac{P_i - g_i}{P_i - \rho_i}\right)\left(\frac{\sigma_i}{g_i}\right) + \Theta_i^{\mathcal{S}}. \tag{3.36}$$

Case 2: When $P_i(t^* - s_j) > \sigma_i + \rho_i(t^* - s_j)$, we get

$$(t^* - s_j) > \frac{\sigma_i}{P_i - \rho_i}. \tag{3.37}$$

From (3.34)

$$(g_i - \rho_i)(t^* - s_j) \leq \sigma_i - g_i D_i^{\mathcal{S}} + g_i \Theta_i^{\mathcal{S}} \tag{3.28}$$

and by substituting for $t^* - s_j$ from (3.37)

$$D_i^{\mathcal{S}} \leq \left(\frac{P_i - g_i}{P_i - \rho_i}\right)\left(\frac{\sigma_i}{g_i}\right) + \Theta_i^{\mathcal{S}}. \tag{3.39}$$

□

A network of $\mathcal{LR}$ servers can be modeled as a single $\mathcal{LR}$ server with latency equal to the sum of the latencies. Thus, the following main result can be derived:

Corollary 1: The maximum delay D_i of a session i in a network of $\mathcal{LR}$ servers, consisting of K servers in series, where the peak rate of the source is known, is bounded as

$$D_i \leq \left(\frac{P_i - g_i}{P_i - \rho_i}\right)\left(\frac{\sigma_i}{g_i}\right) + \sum_{j=1}^{K} \Theta_i^{S_j} \tag{3.40}$$

where $\Theta_i^{S_j}$ is the latency of the jth node.

IV. Schedulers in the $\mathcal{LR}$ Class

In this section, we will show that several well-known work-conserving schedulers belong to the class of $\mathcal{LR}$ servers and determine their latencies. Recall that our definition of $\mathcal{LR}$ servers in the previous section is based on session-busy periods. The following lemma enables determination of an upper bound on the latency of some $\mathcal{LR}$ servers based on its behavior during the session-backlogged periods. We will use this as a tool in our analysis of several well-known schedulers. However, it should be noted that not all schedulers in the $\mathcal{LR}$ class can be analyzed using this approach. Examples of schedulers whose latencies cannot be determined using this approach include all the Rate-Proportional Servers [23], [24] and a class of scheduling algorithms introduced in [25] that perform traffic reshaping.

Lemma 7: Let $(s_j, t_j]$ denote an interval of time during which session i is continuously backlogged in server $\mathcal{S}$. If the service offered to the packets that arrived in the interval $(s_j, t_j]$ can be bounded at every instant t, $s_j < t \leq t_j$ as

$$W_i(s_j, t) \geq \max(0, \rho_i(t - s_j - \Theta_i))$$

then $\mathcal{S}$ is an $\mathcal{LR}$ server with a latency less than or equal to Θ_i.

Note that the converse of this lemma is not true. That is, a finite upper bound for Θ_i may not exist based on the session backlogged period, and the server may still belong to the $\mathcal{LR}$ class. However, Lemma 7 can be used to determine an upper bound on the latency of many well-known work-conserving schedulers. Note that the lemma does not necessarily provide us a tight bound for the parameter Θ_i. An easy way to determine if the bound is tight is to present at least one example in which the offered service is actually equal to the bound. This is the approach we will take in this section to determine the latencies of $\mathcal{LR}$ servers.

Proof of Lemma 7: We will prove the lemma by induction on the number of busy periods. Let us use $(\alpha_k, \beta_k]$ to denote the kth busy period of session i in the server.

Base step: At time α_1, the beginning of the first session-i busy period, the system becomes backlogged for the first time. Let us assume that the first busy period consists of a number of backlogged periods $(s_j, t_j]$. By the definition of session busy period, the following inequality must hold at the beginning of the jth backlogged period:

$$A(\alpha_1, s_j) \geq \rho_i(s_j - \alpha_1).$$

However, the system is empty at time s_j. Therefore,

$$A(\alpha_1, s_j) = W_{i,1}(\alpha_1, s_j). \tag{4.1}$$

Consider any time t in the interval $(s_j, t_j]$. We can write

$$\begin{aligned} W_{i,1}(\alpha_1, t) &= W_{i,1}(\alpha_1, s_j) + W_{i,1}(s_j, t) \\ &\geq \rho_i(s_j - \alpha_1) + \max(0, \rho_i(t - s_j - \Theta_i)) \\ &\geq \max(0, \rho_i(t - \alpha_1 - \Theta_i)). \end{aligned} \tag{4.2}$$

Now consider any time t within the busy period, but between two backlogged periods, i.e., $t_j \leq t \leq s_{j+1}$. Since session i receives no service during the interval $(t, s_{j+1}]$, we can write

$$W_{i,1}(\alpha_1, t) = W_{i,1}(\alpha_1, s_{j+1}). \tag{4.3}$$

But

$$\begin{aligned} W_{i,1}(\alpha_1, s_{j+1}) &= A_i(\alpha_1, s_{j+1}) \\ &\geq \rho_i(s_{j+1} - \alpha_1) \\ &\geq \rho_i(t - \alpha_1). \end{aligned} \tag{4.4}$$

Therefore, from (4.3) and (4.4),

$$W_{i,1}(\alpha_1, t) \geq \rho_i(t - \alpha_1). \tag{4.5}$$

Combining (4.2) and (4.5), we can conclude that, at any instant t during the first busy period $(\alpha_1, \beta_1]$,

$$W_{i,1}(\alpha_1, t) \geq \max(0, \rho_i(t - \alpha_1 - \Theta_i)). \tag{4.6}$$

Inductive step: Assume that the lemma is true for all busy periods $1, 2, \cdots, n$. We will now prove that the lemma is true for the $(n+1)$th busy period as well.

If the system was not backlogged just before the $(n+1)$th busy period started, we can repeat the same proof as in the base step. However, it is possible that, when the $(n+1)$th busy period starts, the system is still backlogged with packets from the nth or earlier busy periods. Let us assume that the backlogged period in progress when the $(n+1)$th busy period starts is the mth backlogged period. This backlogged period started at time s_m; in the general case, we can assume that packets from L earlier busy periods were serviced during the backlogged period $(s_m, t_m]$. Let τ denote the instant at which the last packet from the nth busy period was serviced. Then

$$\begin{aligned} W_i(s_m, \tau) &= \sum_{j=n-L}^{n} A_i(\alpha_j, \beta_j) - W_{i,n-L}(\alpha_{n-L}, s_m) \\ &\leq \rho_i(\beta_n - \alpha_{n-L}) - \rho_i(s_m - \alpha_{n-L}) \\ &= \rho_i(\beta_n - s_m). \end{aligned} \tag{4.7}$$

We need to show that $W_{i,n+1}(\alpha_{n+1}, t) \geq \max(0, \rho_i(t - \alpha_{n+1} - \Theta_i))$. This is trivially satisfied for $t \leq \tau$, since $W_{i,n+1}(\alpha_{n+1}, t) = 0$. For $t > \tau$, we can proceed as follows:

$$\begin{aligned} W_{i,n+1}(\alpha_{n+1}, t) &= W_i(\tau, t) \\ &= W_i(s_m, t) - W_i(s_m, \tau) \\ &\geq \max(0, \rho_i(t - s_m - \Theta_i)) \\ &\quad - \rho_i(\beta_n - s_m) \\ &\geq \max(0, \rho_i(t - \beta_n - \Theta_i)) \\ &\geq \max(0, \rho_i(t - \alpha_{n+1} - \Theta_i)). \end{aligned}$$

The last inequality follows from the fact that $\beta_n < \alpha_{n+1}$ and that the service offered to a session can never be negative. If, after time t, the packets of the $(n+1)$th busy period form multiple backlogged periods, the proof for the base step can be repeated to complete the proof of the lemma. □

Using the above lemma as our tool, we can analyze many well-known servers and prove that they belong to the class of $\mathcal{LR}$ servers and derive their latencies. For lack of space, we illustrate the derivation for only one scheduler, namely PGPS. We refer interested readers to [21], [23] for detailed derivations of the latencies of several schedulers.

To derive the latency of PGPS (Weighted Fair Queueing), we first start by showing that a GPS scheduler is an $\mathcal{LR}$ server.

Lemma 8: A GPS scheduler belongs to the class $\mathcal{LR}$ and its latency is zero.

Proof: From the definition of the GPS multiplexer, during any interval of time that session i is continuously backlogged

$$W_i^{\mathrm{GPS}}(\tau,t) \geq \rho_i(t-\tau).$$

From Lemma 7, it is easy to conclude that a GPS scheduler is an $\mathcal{LR}$ server with zero latency. □

A PGPS (Weighted Fair Queueing) scheduler is the packet-level approximation of GPS multiplexing. In [18] it was proven that, if t^F and t^P are the times that a packet finishes under GPS and PGPS, respectively, then

$$t^P \leq t^F + \frac{L_{\max}}{r}$$

where r is the transmission rate of the outgoing link. For the analysis of a network of $\mathcal{LR}$ servers, it is required that the service be bounded for any time after the beginning of a busy period. In addition, we can only consider that a packet left a packet-by-packet server if all of its bits have left the server. These requirements are necessary in order to provide accurate bounds for the traffic burstiness inside the network. Therefore, just before time t^P the whole packet has not yet departed the packet-by-packet server. Let L_i be the size of the packet. The service offered to session i in the packet-by-packet server will be equal to the service offered to the same session by the GPS server until time t^P minus this last packet. Note also that, since the GPS server has zero latency, the beginning of a busy period finds the system always empty. Therefore,

$$\begin{aligned} W_{i,j}^P(\tau,t) &\geq W_{i,j}^F\left(\tau, t-\frac{L_{\max}}{r}\right) - L_i \\ &\geq \max\left(0, \rho_i\left(t-\tau-\frac{L_{\max}}{r}\right) - L_i\right) \\ &\geq \max\left(0, \rho_i\left(t-\tau-\frac{L_{\max}}{r} - \frac{L_i}{\rho_i}\right)\right). \end{aligned}$$

Hence, we can state the following corollary:

Corollary 2: PGPS is an $\mathcal{LR}$ server with latency

$$\frac{L_{\max}}{r} + \frac{L_i}{\rho_i}.$$

This latency can be shown to be tight.

The latencies of many well-known scheduling algorithms are summarized in Table I. It is easy to see that PGPS, Frame-based Fair Queueing, and VirtualClock all have the lowest latency among all the servers. In addition, their latency does not depend on the number of sessions sharing the same outgoing link. In a self-clocked fair queueing scheduler, however, the latency is a linear function of the maximum number of sessions sharing the outgoing link. In Deficit Round Robin [20], the latency depends on the frame size F. By the definition of the algorithm, the frame size, in turn, is determined by the granularity of the bandwidth allocation and the maximum packet size of its session. That is,

$$\sum_{i=1}^{V} L_i \leq F$$

where L_i is the maximum packet-size of session i. Thus, the latency in Deficit Round Robin is also a function of the number of sessions that share the outgoing link. Weighted Round Robin can be thought of as a special case of Deficit Round Robin and its latency is again a function of the maximum number of sessions sharing the outgoing link.

TABLE I
LATENCIES OF SEVERAL $\mathcal{LR}$ SERVERS

Server	Latency
Generalized Processor Sharing (GPS) [18, 19]	0
Packet-level GPS (PGPS) [18, 19]	$\frac{L_i}{\rho_i} + \frac{L_{max}}{r}$
Self-Clocked Fair Queueing [11]	$\frac{L_i}{\rho_i} + \frac{L_{max}}{r}(V-1)$
VirtualClock [28]	$\frac{L_i}{\rho_i} + \frac{L_{max}}{r}$
Deficit Round Robin [20]	$\frac{(3F-2\phi_i)}{r}$
Weighted Round Robin [16]	$\frac{(F-\phi_i+L_c)}{r}$
Frame-based Fair Queueing [23, 24]	$\frac{L_i}{\rho_i} + \frac{L_{max}}{r}$
Starting Potential-based Fair Queueing [23, 24]	$\frac{L_i}{\rho_i} + \frac{L_{max}}{r}$

V is the number of sessions sharing the outgoing link, L_i the maximum packet size of session I, and $L_{\max}$ the maximum packet size among all the sessions. r is the capacity of the outgoing link and ρ_i The allocated rate of session i. In Weighted Round Robin and Deficit Round Robin, F is the frame size and ϕ_i is the amount of traffic in the frame allocated to session i. L_c is the size of the fixed packet (cell) in Weighted Round Robin.

V. An Improved Delay Bound

The latencies of $\mathcal{LR}$ servers computed in the last section are based on the assumption that a packet is considered serviced only when its last bit has left the server. Thus, the latency Θ_i^S was computed such that the service offered during a session busy period during an interval $(\tau,t]$, denoted by $W_{i,j}(\tau,t)$, is always greater than or equal to $\rho_i(t-\tau-\Theta_i^S)$. Since the maximum difference between $W_{i,j}(\tau,t)$ and $\rho_i(t-\tau)$ occurs just before a session-i packet departs the system, the latency Θ_i^S is calculated at such points. This is necessary to be able to bound the arrivals to the next server in a chain of servers; since our servers are not cut-through devices, a packet can be serviced only after its last bit has arrived. Our assumption that the packet leaves as an impulse from the server allows us to model the arrival of the packet in the next server as an impulse as well.

When we compute the end-to-end delay of a session, however, we are only interested in finding the time at which the last bit of a packet leaves the last server. Thus, for the last server in a chain, we can determine the latency Θ_i^S based only on the instants of time just after a packet was serviced from the session. This results in a lower value of latency and, consequently, a tighter bound on the end-to-delay in a network of servers than that given by (3.28).

To apply this idea, the analysis of the network is separated into two steps. If the session passes through k hops, we bound the service offered to the session in the first $k-1$ servers considering arbitrary instants during session-busy periods. In the last node, however, we calculate the latency based only on the points just after a packet completes service.

This idea is best illustrated by an example in the case of the PGPS server. Let $W_{i,j}^P(\tau, t)$ denote the service received by session i during an interval $(\tau, t]$ within its jth busy period in the PGPS server, and let $W_{i,j}^F(\tau, t)$ denote the same parameter in the corresponding GPS server. Assume that a busy period starts at time τ and that a packet leaves the PGPS server at time t_k. Then, on the corresponding GPS server, this packet left at time $t_k - L_{\max}/r$ or later. Therefore, if we consider only such points t_k, we can write

$$\begin{aligned} W_{i,j}^P(\tau, t_k) &\geq W_{i,j}^F\left(\tau, t_k - \frac{L_{\max}}{r}\right) \\ &\geq \rho_i\left(t_k - \tau - \frac{L_{\max}}{r}\right). \end{aligned}$$

This results in a latency of $L_{\max}/r$ as compared to $(L_i/\rho_i + L_{\max}/r)$ obtained by considering all points during the session busy period.

Fig. 4 shows the two envelopes based on bounding the service received by the session in two different ways. The lower envelope applies to arbitrary points in the session-busy period, while the upper envelope is valid only at points when a packet leaves the system. For computing end-to-end delay bounds in a network of servers, we can use the upper envelope in the last server. In all the work-conserving schedulers we have studied, the two envelopes are apart by L_i/ρ_i, where L_i is the maximum packet-size for session i. Therefore, for these $\mathcal{LR}$ servers, we can obtain an improved bound for the end-to-end delay in a network by subtracting L_i/ρ_i from (3.28). Therefore,

$$D_i \leq \frac{\sigma_i}{\rho_i} + \sum_{j=1}^{k} \Theta_i^{(S_j)} - \frac{L_i}{\rho_i}. \tag{5.1}$$

Let us assume a network of N PGPS servers, where rate g_i^k is allocated to session i in server k. If we substitute the latency

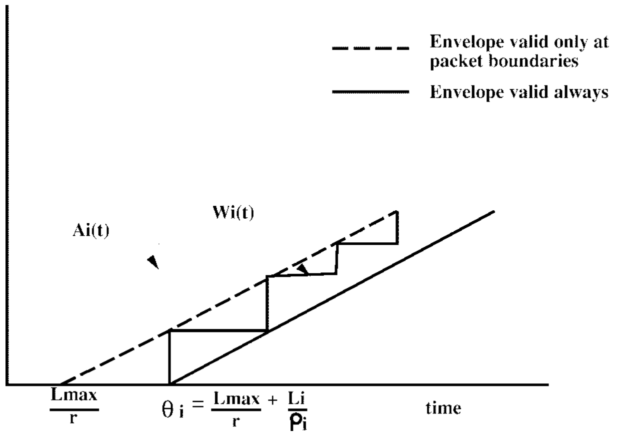

Fig. 4. Illustration of the two envelopes used to bound the service received by session i in a session busy period. Each step in the arrival function indicates a new packet. The lower envelope is a valid lower bound for the service at any point in the busy period, while the upper one is valid only at the points when a packet leaves the system.

obtained for PGPS from Table I in the above expression, that is, $\Theta_i^{(S_j)} = L_i/g_i + L_{\max}/r$, we get

$$D_i \leq \frac{\sigma_i}{\min\limits_{k=i,2,\cdots,N} g_i^k} + \sum_{k=1}^{N-1} \frac{L_i}{g_i^k} + N\frac{L_{\max}}{r}. \tag{5.2}$$

Note that this bound is tighter than the one reported by Parekh and Gallager [19] for a network of PGPS servers. Since the latencies of PGPS and VirtualClock are identical, the bound of (5.2) applies to VirtualClock as well; this is also tighter than the one presented in [17].

While we have verified that this improvement of L_i/ρ_i in the delay bound is valid for all the $\mathcal{LR}$ servers analyzed in this paper, whether this is true for all $\mathcal{LR}$ servers remains an open question. We have not yet found a formal proof on its validity for arbitrary $\mathcal{LR}$ servers.

Effect of Propagation Delay

In the previous section, we assumed that the propagation delays in the network are negligible. It is easy to verify that the analysis is still valid when the propagation delays are greater than zero. When the end-to-end propagation delay is constant, it can easily be incorporated in the latency parameter. Thus, if p_i^j is the propagation delay between nodes j and $j+1$, the delay bound becomes

$$D_i \leq \frac{\sigma_i}{\rho_i} + \sum_{j=1}^{k} \Theta_i^{(S_j)} - \frac{L_i}{\rho_i} + \sum_{j=1}^{k-1} p_i^j. \tag{5.3}$$

VI. Related Work

Our approach in modeling the worst-case behavior of scheduling algorithms with respect to an end-to-end session is related to the work of Cruz [2]–[6], Zhang [27], and Parekh and Gallager [18], [19]. Cruz [2], [3] analyzed the end-to-end delay, buffer requirements, and internal network burstiness of sessions in an arbitrary topology network where all sources are token-bucket-controlled. While the objectives of our analysis are similar, there are three major differences between the

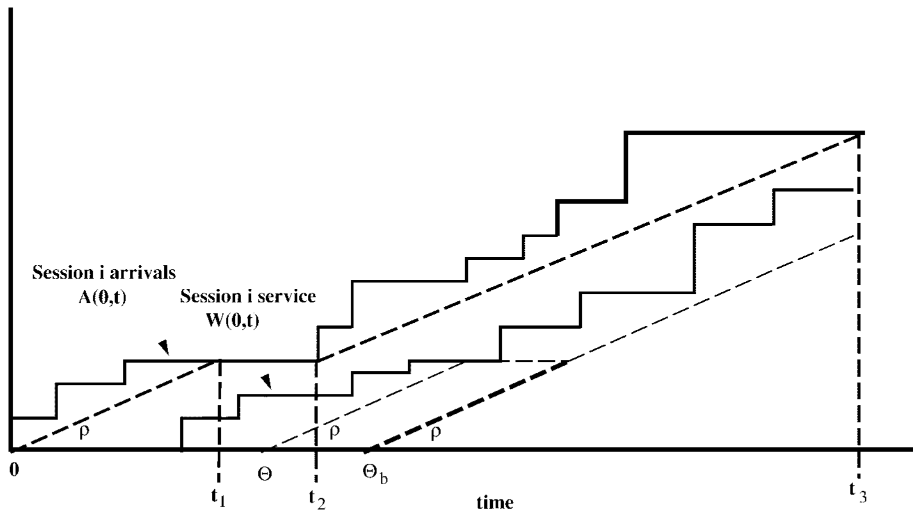

Fig. 5. Difference in bounding service based on the backlogged and busy periods. $(0, t_1)$ and (t_2, t_3) are two consecutive busy periods of a session. Bounding the service during each busy period with a linear envelope results in a latency of Θ, while bounding the same service during the backlogged period $(0, t_3)$ results in a latency of Θ_b.

approaches taken. First, the class of scheduling algorithms we study are capable of providing bandwidth guarantees to individual sessions. Therefore, we can derive deterministic end-to-end delay guarantees that are independent of the behavior of other sessions. Second, we do not study individual schedulers in isolation and accumulate the session delays as in [2], [3], but instead model the behavior of the chain of schedulers on the path of the session as a whole. Third, we estimate the latency parameters for the individual schedulers tightly, taking into account their internal structure. Thus, our approach, in general, provides much tighter end-to-end delay bounds for individual sessions.

Parekh and Gallager analyzed the worst-case behavior of sessions in a network of GPS schedulers [18], [19] and derived upper bounds on end-to-end delay and internal burstiness of sessions. However, the analysis applies to a homogeneous network consisting of only GPS schedulers. Our analysis accommodates a broad range of scheduling algorithms and the ability to combine the schedulers in arbitrary ways in a network.

Zhang [27] derived end-to-end delay bounds for a class of nonwork-conserving scheduling algorithms when traffic is reshaped at each node of the network. This allows the delays of individual schedulers on the path to be accumulated in a simple manner. Our approach differs from this work in that we consider the broader class of work-conserving schedulers in our analysis, and we do not assume any traffic reshaping mechanisms within the network.

An important contribution of the theory of $\mathcal{LR}$ servers is the notion of the busy period. The bound on the service offered by an $\mathcal{LR}$ server is based on the busy period and is independent of the periods of time that the server is backlogged. This is a more general approach than bounding the service offered by the server based on the backlogged period. An approach based on the latter was proposed by Cruz [5], [6]. This model bounds the service offered to a session during one or more backlogged periods, thus providing a means to design a class of scheduling algorithms that can provide specific end-to-end delay guarantees. Using the concept of busy period instead of backlogged period in this model can lead to tighter end-to-end delay bounds and a larger class of schedulers that can provide these delay bounds. This is illustrated by the following example.

In Lemma 7, we proved that, if we use the backlogged period to bound the service offered by a server $\mathcal{S}$, then $\mathcal{S}$ is an $\mathcal{LR}$ server and its latency cannot be larger than that found for the backlogged period. However, we must emphasize the fact that the opposite is not true. Consider the example of Fig. 5. Let us assume a session i in an $\mathcal{LR}$ server with allocated rate ρ and latency Θ. Referring to Fig. 5, time intervals $(0, t_1]$ and $(t_2, t_3]$ form two busy periods. However, the server remains backlogged during the whole interval $(t_1, t_3]$. If the backlogged period was used to bound the service offered by the server, a latency $\Theta_b > \Theta$ would result. By extending the above example over multiple busy periods, it is easy to verify that Θ_b may not even be bounded. This shows that if we were to bound the service during a backlogged period in an $\mathcal{LR}$ server with a linear envelope, the x-intercept of the envelope could be much larger, and may even be infinite. Thus, the resulting end-to-end delay bounds can be much higher, and may not even be bounded. Therefore, the class of $\mathcal{LR}$ servers is larger than the class of servers based on bounds derived with respect to the backlogged period. Since the first presentation of this work in [22], the notion of the busy period has been extended to nonlinear service curves in [1], [7].

The $\mathcal{LR}$ model is also closely related to the work of Cruz [4], where the delay behavior of a scheduler is characterized in terms of its *service burstiness*. The service burstiness measure is similar to the latency parameter in the $\mathcal{LR}$ model and allows bounds on delays to be obtained for a class of scheduling algorithms from their service burstiness parameters and the traffic parameters of the session under consideration. The model enables multiple schedulers in series to be modeled by a single scheduler by combining the service burstiness parameters of the individual schedulers, thus providing much tighter bounds on end-to-end delay than those obtained by combining the delay bounds of individual schedulers. However, the service burstiness is still defined in terms of the behavior of the scheduler during a backlogged period. Therefore, as shown in the previous paragraph, the resulting delay bounds may not be as tight as those obtained by the $\mathcal{LR}$ model.

Hung and Kesidis [14] presented a method for computation of delay bounds in schedulers. Their results are restricted to isolated packets in a single scheduler and the extension to arbitrary traffic distributions and networks of servers is not straightforward. In addition, their model is based on the virtual finishing times of packets in the scheduler, which makes it difficult to apply the model to other scheduler architectures, such as round-robin schedulers. Our approach, in contrast, is independent of the scheduler architecture and applies to diverse models of session traffic and arbitrary networks of schedulers. Another model for delay-analysis based on a class of guaranteed-rate servers was presented in [13]. A possible limitation of this model, however, is that it is closely coupled with timestamp-based algorithms; the analysis of scheduling algorithms based on other architectures is not straightforward. The $\mathcal{LR}$ class provides a more natural approach for analyzing the worst-case behavior of traffic-scheduling algorithms, independent of the scheduler architecture. Finally, Golestani recently presented a delay analysis of a class of fair-queueing algorithms including Self-Clocked Fair Queueing [12]. However, this analysis does not apply to unfair algorithms such as VirtualClock.

VII. Conclusions

In this paper, we presented a general model for analyzing the worst-case end-to-end delay behavior of a session in a network of servers, where each node in the network can be characterized as a Latency-Rate server. We presented a general framework for analyzing these networks and provided the tools for classifying scheduling algorithms as $\mathcal{LR}$ servers. This methodology enables the delay properties of a scheduling algorithm to be characterized in terms of a single *latency* parameter, thus allowing end-to-end delay bounds to be computed without detailed knowledge of the algorithm.

In Table I, we summarized the latencies of several scheduling algorithms belonging to the $\mathcal{LR}$ class. Based on this summary, it is easy to see that schedulers such as VirtualClock and Frame-based Fair Queueing have the same delay properties as that of PGPS. Indeed, in [23], we have shown that this is the case for a whole class of algorithms.

It should be noted that our approach of modeling a network of $\mathcal{LR}$ servers as a single $\mathcal{LR}$ server did not also make any assumptions about the input traffic. Although the bounds derived in this paper for the end-to-end delays and buffer requirements are for specific traffic models, the same $\mathcal{LR}$ model can be used for analysis with other models for bounding session arrivals. For example, it is straightforward to derive delay bounds in a network of $\mathcal{LR}$ servers based on the *Exponentially-Bounded-Burstiness* model of arrivals [26], [29].

In closing, we must note that it can be misleading to compare scheduling algorithms based solely on their latencies. The criteria used for distribution of free bandwidth among the sessions is also important. Two different methods of distributing excess bandwidth were presented in [23] and [9]. However, the end-to-end delay bounds do not depend on how the scheduler distributes the excess bandwidth. Indeed, in the calculation of our worst-case bounds, we made no assumptions about this aspect of the schedulers.

Acknowledgment

The authors thank Prof. G. Sasaki for reading the manuscript carefully and suggesting numerous revisions and improvements. The anonymous reviewers also provided constructive comments and criticism.

References

[1] R. Agrawal and R. Rajan, "Performance bounds for guaranteed and adaptive services," Tech. Rep. RC 20649, IBM Research Div., 1996.

[2] R. L. Cruz, "A calculus for network delay. I. Network elements in isolation," *IEEE Trans. Inform. Theory*, vol. 37, pp. 114–131, Jan. 1991.

[3] ——, "A calculus for network delay. II. Network analysis," *IEEE Trans. Inform. Theory*, vol. 37, pp. 132–141, Jan. 1991.

[4] ——, "Service burstiness and dynamic burstiness measures: A framework," *J. High-Speed Networks*, vol. 1, no. 2, pp. 105–127, 1992.

[5] R. L. Cruz and H. N. Liu, "End-to-end queueing delay in ATM networks," *J. High Speed Networks*, vol. 2, no. 3, pp. 149–164, 1994.

[6] R. L. Cruz, "Quality of service guarantees in virtual circuit switched networks," *IEEE J. Select. Areas Commun.*, vol. 13, pp. 1048–1056, Aug. 1995.

[7] R. L. Cruz and C. M. Okino, "Service guarantees for window flow control," in *Proc. 34th Allerton Conf. on Communication Control and Computing*, Monticello, IL, Oct. 1996, pp. 10–21.

[8] A. Demers, S. Keshav, and S. Shenker, "Analysis and simulation of a fair queueing algorithm," *Internetworking: Research and Experience*, vol. 1, no. 1, pp. 3–26, 1990.

[9] N. Duffield, T. V. Lakshman, and D. Stiliadis, "On adaptive bandwidth sharing with rate guarantees," In *Proc. IEEE INFOCOM'98*, San Francisco, CA, Apr. 1998.

[10] S. J. Golestani, "A framing strategy for congestion management," *IEEE J. Select. Areas Commun.*, vol. 9, pp. 1064–1077, Sept. 1991.

[11] ——, "A self-clocked fair queueing scheme for broadband applications," in *Proc. IEEE INFOCOM '94*, Toronto, ON, Canada, Apr. 1994, pp. 636–646.

[12] ——, "Network delay analysis of a class of fair queueing algorithms," *IEEE J. Select. Areas Commun.*, vol. 13, pp. 1057–1070, Aug. 1995.

[13] P. Goyal, S. L. Lam, and H. M. Vin, "Determining end-to-end delay bounds in heterogeneous networks," in *Proc. 5th Int. Workshop on Network and Operating System Support for Digital Audio and Video*, Durham, NH, Apr. 1995, pp. 287–298.

[14] A. Hung and G. Kesidis, "Bandwidth scheduling for wide-area ATM networks using virtual finishing times," *IEEE/ACM Trans. Networking*, vol. 4, pp. 49–54, Feb. 1996.

[15] C. R. Kalmanek, H. Kanakia, and S. Keshav, "Rate controlled servers for very high-speed networks," in *Proc. IEEE Global Telecommunications Conf.*, Dec. 1990, pp. 300.3.1–300.3.9.

[16] M. Katevenis, S. Sidiropoulos, and C. Courcoubetis, "Weighted round-robin cell multiplexing in a general-purpose ATM switch chip," *IEEE J. Select. Areas Commun.*, vol. 9, pp. 1265–1279, Oct. 1991.

[17] S. S. Lam and G. G. Xie, "Burst scheduling: Architecture and algorithm for switching packet video," in *Proc. INFOCOM'95*, Boston, MA, Apr. 1995, pp. 940–950.

[18] A. K. Parekh and R. G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: The single-node case," *IEEE/ACM Trans. Networking*, vol. 1, no. 3, pp. 344–357, June 1993.

[19] ——, "A generalized processor sharing approach to flow control in integrated services networks: The multiple node case," *IEEE/ACM Trans. Networking*, vol. 2, no. 2, pp. 137–150, Apr. 1994.

[20] M. Shreedhar and G. Varghese, "Efficient fair queueing using deficit round-robin," *IEEE/ACM Trans. Networking*, vol. 4, no. 3, pp. 375–385, June 1996.

[21] D. Stiliadis, “Traffic scheduling in packet-switched networks: Analysis, design and implementation,” Ph.D. dissertation, Univ. of California, Santa Cruz, 1996. [Online]. Available WWW: www.cse.ucsc.edu/research/hsnlab/publications/
[22] D. Stiliadis and A. Varma, “Latency-rate servers: A general model for analysis of traffic scheduling algorithms,” in *Proc. IEEE INFOCOM ’96*, San Francisco, CA, Apr. 1996, pp. 111–119.
[23] ——, “Rate-proportional servers: A design methodology for fair queueing algorithms,” *IEEE/ACM Trans. Networking*, vol. 6, no. 2, pp. 164–174, Apr. 1998.
[24] ——, “Efficient fair queueing algorithms for packet switched networks,” *IEEE/ACM Trans. Networking*, vol. 6, no. 2, pp. 175–185, Apr. 1998.
[25] ——, “A general methodology for designing efficient traffic scheduling and shaping algorithms,” in *Proc. IEEE INFOCOM’97*, Kobe, Japan, Apr. 1997, pp. 326–335.
[26] O. Yaron and M. Sidi, “Performance and stability of communication networks via robust exponential bounds,” *IEEE/ACM Trans. Networking*, vol. 1, no. 3, pp. 372–385, June 1993.
[27] H. Zhang, “Service disciplines for packet-switching integrated services networks,” Ph.D. dissertation, Univ. of California, Berkeley, 1992.
[28] L. Zhang, “VirtualClock: A new traffic control algorithm for packet switching networks,” *ACM Trans. Computer Systems*, vol. 9, no. 2, pp. 101–124, May 1991.
[29] Z. Zhang, D. Towsley, and J. Kurose, “Statistical analysis of generalized processor sharing scheduling discipline,” in *Proc. ACM SIGCOMM ’94*, Sept. 1994, pp. 68–77.

Dimitrios Stiliadis, for photograph and biography, see p. 174 of the April 1998 issue of this TRANSACTIONS.

Anujan Varma (M’86), for photograph and biography, see p. 174 of the April 1998 issue of this TRANSACTIONS.