Design of a Stochastic Traffic Regulator for End-to-End Network Delay Guarantees

Massieh Kordi Boroujeny, *Student Member, IEEE*, and Brian L. Mark, *Senior Member, IEEE*

***Abstract*—Providing end-to-end network delay guarantees in packet-switched networks such as the Internet is highly desirable for mission-critical and delay-sensitive data transmission, yet it remains a challenging open problem. Since deterministic bounds are based on the worst-case traffic behavior, various frameworks for stochastic network calculus have been proposed to provide less conservative, probabilistic bounds on network delay, at least in theory. However, little attention has been devoted to the problem of regulating traffic according to stochastic burstiness bounds, which is necessary in order to guarantee the delay bounds in practice. We design and analyze a stochastic traffic regulator that can be used in conjunction with results from stochastic network calculus to provide probabilistic guarantees on end-to-end network delay. Two alternative implementations of the stochastic regulator are developed and compared. Numerical results are provided to demonstrate the performance of the proposed stochastic traffic regulator.**

***Index Terms*—Stochastic network calculus, traffic shaper, end-to-end delay, traffic burstiness bounds.**

I. Introduction

Currently, the Internet does not provide end-to-end delay guarantees for traffic flows. Even if the path taken by a given traffic flow is fixed, e.g., via mechanisms such as software-defined networking or multi-protocol label switching, network congestion arising from other flows can result in highly variable delays. The variability and random nature of traffic flows in a packet-switched network make it very challenging to provide performance guarantees. End-to-end delay guarantees would improve the user experience provided by real-time applications such as video streaming and video conferencing, as well as enable emerging applications involving virtual reality and augmented reality. Bounds on network delay are of particular relevance to age of information (AoI), a performance metric whereby the freshness of data is assessed from the receiver's perspective [2].

The standard approach to providing network performance guarantees consists of two basic elements:

1) *Admission control:* A new flow is admitted only if performance guarantees can be maintained for all admitted flows with the available network resources.
2) *Traffic regulation:* Each admitted traffic flow is regulated to ensure that it does not consume more resources than what was negotiated by the admission control scheme.

Admission control relies on a means of characterizing the traffic to determine how to allocate network resources to the new flow. The random and bursty nature of traffic flows in packet-switched networks make them difficult to characterize. Even if each traffic flow is modeled as a random arrival process, e.g., a Markov modulated Poisson process, the problem of resource allocation to provide end-to-end performance guarantees is intractable. Moreover, traffic regulation to ensure that a flow conforms to the parameter of a traffic model is infeasible.

In his seminal work, Cruz [3], [4] proposed the (σ, ρ) characterization of traffic, which imposes a deterministic bound on the burstiness of a traffic flow. The bound ensures that the long-term average arrival rate a (σ, ρ)-bounded traffic source does not exceed the rate parameter ρ and its maximum burst size does not exceed the burst size parameter σ. By bounding traffic flows according to (σ, ρ) parameters, Cruz developed a network calculus which determined how these parameters propagate through network elements and derived associated end-to-end delay bounds.

An important feature of the (σ, ρ) characterization is that it can be enforced by a traffic regulator. In fact, the (σ, ρ) traffic bound can be defined operationally in terms of a (σ, ρ) traffic regulator. For bursty traffic sources, however, the (σ, ρ) bound can lead to worst-case end-to-end delay bounds that are very conservative, which in turn results in very low network resource utilization. To achieve higher utilization, the approach in [5] estimates a (σ, ρ)-based parameter for an arbitrary traffic source by minimizing a cost function derived from the concept of effective bandwidths [6, Chap. 9], subject to a constraint on the shaping delay incurred on the source. Given the traffic parameter, worst-case delay bounds for a traffic source could be computed using deterministic network calculus. Alternatively, resource allocation could be performed using effective bandwidths, but in this case true performance guarantees would not be provided.

The (σ, ρ) traffic bound of Cruz was the basis for further research into stochastic bounds on traffic burstiness and stochastic network calculus to provide probabilistic end-to-end delay guarantees as opposed to worst-case delay bounds. Stochastic network calculus remains an active topic of research [7]. To our knowledge, however, a traffic regulator to enforce a *stochastic* traffic bound based on stochastic network calculus has not been addressed previously, despite the fact that such a regulator is necessary to provide performance guarantees in real networks. Stochastic network calculus is based on the

Manuscript received August 19, 2020; revised July 30, 2021 and March 23, 2022; accepted May 3, 2022; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor L. Huang. This work was supported in part by the U.S. National Science Foundation under Grant No. 1717033. A preliminary version of this work was presented at the IEEE Int. Conf. on Communications (ICC'2020) [1]. *(Corresponding author: Brian L. Mark.)*

The authors are with the Dept. of Electrical and Computer Engineering, George Mason University, Fairfax, VA 22030 (e-mail: mkordibo@gmu.edu, bmark@gmu.edu).

Digital Object Identifier 10.1109/TNET.2022.3181858

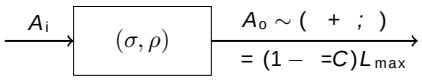


Fig. 1. (;) regulator with input/output links of capacity C.

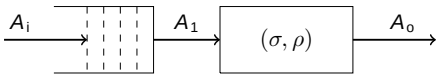


Fig. 2. (;) traffic shaper with front-end buffer.

assumption that all traffic flows entering the network satisfy stochastic traffic bounds. In this paper, we develop a traffic regulator to enforce the so-called *generalized Stochastically Bounded Burstiness* (gSBB) traffic burstiness bound in [8], [9]. The gSBB bound is closely related to the SBB and EBB bounds in [10] and [11], respectively. Formal definitions of these bounds are given in Section II-B. These bounds are also related to the moment generating function (MGF) traffic bounds discussed in [6], [12]. We focus on the gSBB bound primarily because it is more amenable to traffic regulation than the other traffic bounds (see Section IV).

We refer to our proposed regulator as a stochastic $(\sigma \; , \rho)$ regulator, since the burst size parameter σ varies over time and takes on values in a finite set $\Sigma = f\sigma_1, \ldots, \sigma_M g$. We describe the design and basic properties of the stochastic $(\sigma \; , \rho)$ regulator and develop two practical implementations. Our analysis establishes that the output traffic always conforms to the gSBB bound.

The remainder of the paper is organized as follows. In Section II, we review basic concepts in deterministic and stochastic network calculus. In Section III, we review key properties of the deterministic (σ, ρ) regulator and develop some new results for its analysis, which are applied in Section IV to the design and implementation of the proposed stochastic $(\sigma \; , \rho)$ regulator. Numerical results demonstrating the performance of the $(\sigma \; , \rho)$ regulator are presented in Section V. Concluding remarks are given in Section VI.

II. Background on Network Calculus

Our proposed stochastic traffic regulator builds on the (σ, ρ) network calculus of Cruz [3], [4] as well as stochastic network calculus. In this section, we review relevant aspects of both types of network calculus.

A. Deterministic (σ, ρ) Network Calculus

Let $A = fA(t) : t \quad 0g$ denote a traffic process or flow, where $A(t)$ represents the amount of traffic arriving in the interval $[0, t)$. We shall assume that t is a continuous-time parameter, although our results carry over to the discrete-time case as well.

Definition 1 ((σ, ρ) traffic bound). A traffic flow A is said to be (σ, ρ)-bounded, denoted as $A \quad (\sigma, \rho)$, if

$$A(t) \quad A(s) \quad \rho(t \quad s) + \sigma, \quad 8s \; 2 \; [0, t], \tag{1}$$

where $\sigma, \rho \quad 0$.

In conjunction with Definition 1, Cruz [3] introduced a traffic regulator to enforce conformance to the (σ, ρ) bound. For an idealized fluid model of input traffic, a (σ, ρ) traffic regulator ensures that the *output* traffic $A_o \quad (\sigma, \rho)$ and traffic departs the regulator in the same order as it arrives to the regulator, i.e., the service discipline is first-come first-served (FCFS). When the traffic consists of discrete packets of maximum length L_{max} and the input/output links to the regulator have finite capacity C, the output traffic satisfies $A_o \quad (\sigma + \delta, \rho)$, where (see Fig. 1)

$$\delta = (1 \quad \rho/C) L_{max}. \tag{2}$$

Traffic regulation can be accomplished by packet dropping, packet tagging (as lower priority), or delaying of packets. In the first two cases, the traffic regulator is sometimes referred to as a *traffic policer* whereas in the third case it is referred to as a *traffic shaper*. The traffic regulators discussed in this paper will be of the traffic shaper variety. A traffic shaper includes a front-end buffer, which stores packets that are delayed in the process of forcing the output traffic to conform to (σ, ρ) (see Fig. 2). A traffic policer is equivalent to a traffic shaper with no front-end buffer; i.e., packets that do not conform to (σ, ρ) are dropped or tagged immediately rather than placed in a buffer.

B. Stochastic Network Calculus

The (σ, ρ) bound in (1) tends to be very conservative for bursty traffic. This is illustrated in Fig. 12 (see Section V) for a sample path of bursty traffic fed to a queue with service rate ρ. The queue size reaches the burstiness bound σ, but is far below σ most of the time. End-to-end delay bounds based on worst-case (σ, ρ) bounds will be overly conservative for bursty traffic flows. Admission control based on such bounds will lead to low network utilization. Moreover, deterministic network calculus cannot exploit the phenomenon of statistical multiplexing. These considerations motivated the development of *stochastic* traffic burstiness bounds and stochastic network calculus to allow the derivation of stochastic end-to-end delay bounds [7], [13].

An early proposal for a stochastic traffic burstiness bound was the Exponentially Bounded Burstiness (EBB) of Yaron and Sidi [14], which involves an exponential bounding function. A related traffic bound based on moment generating functions was proposed by Chang [12]. In this paper, we focus on the *generalized* Stochastically Bounded Burstiness (gSBB) proposed in [9].

Definition 2 (gSBB). A traffic process A is gSBB with upper rate ρ and bounding function $f \; 2 \; BF$ if

$$\mathsf{P} f W \; (t; A) \quad \sigma g \quad f(\sigma), \quad 8t \quad 0, \; 8\sigma \quad 0, \tag{3}$$

where BF denotes the family of positive non-increasing real-valued functions and $W \; (t; A)$ is the virtual workload at time t of an infinite-buffer FCFS (First Come First Served) queue with constant service rate ρ with input traffic A. The virtual workload is given by

$$W \; (t; A) = \max_{0 \quad s \quad t} [A(t) \quad A(s) \quad \rho(t \quad s)]. \tag{4}$$

Intuitively, the virtual workload at an arbitrary time t is the amount of work (e.g., in units of bits) remaining in the system for the server to process. The gSBB concept is based on Stochastically Bounded Burstiness (SBB) [10].

Definition 3 (SBB). A traffic process A is SBB with upper rate ρ and bounding function $f \in \mathcal{F}$ if

$$\mathbb{P}\{A(t) - A(s) - \rho(t-s) - \sigma\} \leq f(\sigma), \quad \forall t \geq 0, \forall \sigma \geq 0, \tag{5}$$

where $\mathcal{F}$ is the family of functions f such that for every $n, \sigma \geq 0$, the n-fold integral $\int_0^{\sigma} du \int_0^{u} \cdots \int_0^{u} f(u)$ is bounded.

A traffic process is EBB if it is SBB with an exponentially decaying bounding function, i.e., $f(\sigma) = ae^{-\alpha\sigma}$, where $a, \alpha > 0$. For a given SBB bounding function $f \in \mathcal{F}$, a traffic process that is gSBB with respect to f is also SBB. Thus, the gSBB bound is more conservative than the SBB bound. Nevertheless, the gSBB concept has two important advantages over SBB, which we leverage in designing the (σ, ρ) regulator (see Section IV): 1) $\mathcal{BF} \subset \mathcal{F}$; 2) The gSBB bound is defined in terms of $W(t; A)$ rather than $A(t)$.

The phase-type traffic bound proposed in [15] is closely related to gSBB.

Definition 4. A traffic process A is characterized by a *phase-type traffic descriptor* $[\rho; (a, \mathbf{\Lambda}, \mathbf{Q}, T)]$ if

$$\mathbb{P}\{W(t; A) - \sigma\} \leq a\, e^{\mathbf{Q}\sigma}\mathbf{1}, \tag{6}$$

for all $t \geq 0$ and all $\sigma \in (0, T]$. Here, $\mathbf{1}$ is a column vector of all ones, $a \geq 0$, $T > 0$, and $(\mathbf{\Lambda}, \mathbf{Q})$ denotes the parameter of a phase-type distribution [16], [17].

When $T = \infty$, the phase-type traffic bound is a particular case of gSBB. Using Algorithm 1 in [18], any given traffic flow can be characterized by a phase-type traffic descriptor.

Analogous to the deterministic network calculus, a stochastic network calculus can be developed based on a given stochastic traffic burstiness bound [6], [10], [14]. By applying results from the stochastic network calculus based on gSBB (see [9]), the admissibility of a given set of traffic flows with respect to a certain probabilistic end-to-end delay constraint can be determined. More general stochastic traffic bounds have since been developed in conjunction with notions of statistical arrival envelopes, service curves, and min-plus algebra in the context of stochastic network calculus [7]. However, end-to-end delay guarantees via stochastic network calculus can only be provided if the user traffic flows that are offered as input to the network conform to their negotiated traffic burstiness bounds. The stochastic traffic regulator developed in this paper can be applied at the network edge to ensure that a user's traffic does not violate the gSBB bound provided to the admission control unit. Additional performance benefits can be obtained by applying stochastic traffic regulation in internal network elements to reshape traffic flows to their negotiated parameters, since the traffic parameter of a flow may be altered after it passes through a network element.

III. Analysis of Deterministic (σ, ρ) Regulator

The (σ, ρ) regulator may be viewed as an extension of the deterministic (σ, ρ) regulator, in which the burst size parameter σ takes on values from a finite set $\Sigma = \{\sigma_1, \ldots, \sigma_M\}$ while adapting to the input traffic flows. In particular, the operation of a (σ, ρ) regulator can be viewed as a special case of a (σ, ρ) regulator. In Section III-A, we review some relevant results on the virtual workload process $W(t)$ from the (σ, ρ) calculus. In Section III-B, we develop some new results related to $W(t)$, which we shall use in the design and analysis of the stochastic (σ, ρ) regulator in Section IV.

A. Input/Output Workload Analysis

Suppose a traffic flow A is offered to an infinite-buffer FCFS system with constant service rate ρ. Clearly, the virtual workload $W(t; A)$ is a decreasing function of ρ. It can easily be shown that $A \sim (\sigma, \rho)$ if and only if

$$W(t; A) \leq \sigma, \quad \forall t \geq 0. \tag{7}$$

Equation (7) provides a useful alternative characterization of a (σ, ρ)-bounded traffic flow.

Now suppose that the input and output traffic links to and from a (σ, ρ) regulator have a finite capacity $C > \rho$. Consider an input traffic flow A_{i} to the regulator. Let s_j denote the arrival time of the jth packet, t_j its departure time, and L_j its length in bits. The jth packet begins arriving at time s_j and is received completely at the regulator at time $a_j = s_j + L_j/C$. We assume that a packet does not arrive when the previous one is being received. i.e., $a_j < s_{j+1}$.

The operation of the regulator can be described in terms of the workload $W(s_j; A_{\mathsf{i}})$. At time s_j, if $W(s_j; A_{\mathsf{i}}) > \sigma$, the regulator delays the packet such that at its departure time t_j, the condition $W(t_j; A_{\mathsf{o}}) \leq \sigma$ holds. Hence, the departure time of the jth packet is derived as [3]

$$t_j = [W(s_j; A_{\mathsf{i}}) - \sigma]^+/\rho + s_j, \tag{8}$$

where $[x]^+ := \max\{x, 0\}$. The packet completely departs the regulator at time

$$b_j = t_j + L_j/C. \tag{9}$$

At times other than departures, the workload may not necessarily be bounded by σ, but always satisfies [3]

$$W(t; A_{\mathsf{o}}) \leq \sigma + (1 - \rho/C)L_{\mathsf{max}}, \quad \forall t \geq 0. \tag{10}$$

Thus, $A_{\mathsf{o}} \sim (\sigma + \delta, \rho)$, where δ, given by (2), can be viewed as the maximum error margin in regulating packetized traffic when the input/output links have capacity C (see Fig. 1).

As shown Fig. 3, when a packet is being received by the regulator, e.g., during $[s_j, a_j]$, the workload $W(t; A_{\mathsf{i}})$ increases linearly with slope $C - \rho$. Conversely, during the time between the complete arrival of a packet and the initial arrival of the next packet to the system, e.g., during $[a_j, s_{j+1}]$, the workload $W(t; A_{\mathsf{i}})$ decreases linearly with slope $-\rho$. Similarly, when a packet departs the regulator, e.g., during $[t_j, b_j]$, the workload $W(t; A_{\mathsf{o}})$ increases linearly with slope $C - \rho$. When packets are not departing the system, e.g., during $[b_j, t_{j+1}]$, $W(t; A_{\mathsf{o}})$ decreases linearly with slope $-\rho$. Assume that the buffer of the regulator is empty at $t = s_1$.

Let

$$\delta_j = (1 \quad \rho/C)L_j \tag{11}$$

denote the error margin due to regulating the jth packet. We present the governing equations for a (σ, ρ) regulator in terms of the workloads $W\ (t; A_{\mathsf{i}})$ and $W\ (t; A_{\mathsf{o}})$ as follows:

$$W\ (t; A_{\mathsf{i}}) = [W\ (a_{j\ \ 1}; A_{\mathsf{i}}) \quad \rho(t \quad a_{j\ \ 1})]^+, \quad t\ \mathit{2}\ [a_{j\ \ 1}, s_j]; \tag{12}$$

$$W\ (t; A_{\mathsf{i}}) = W\ (s_j; A_{\mathsf{i}}) + (t \quad s_j)(C \quad \rho),\ t\ \mathit{2}\ [s_j, a_j]; \tag{13}$$

$$W\ (t_j; A_{\mathsf{o}}) = \begin{array}{ll} \sigma, & \text{if } W\ (s_j; A_{\mathsf{i}}) > \sigma, \\ W\ (s_j; A_{\mathsf{i}}), & \text{if } W\ (s_j; A_{\mathsf{i}}) \quad \sigma; \end{array} \tag{14}$$

$$W\ (t; A_{\mathsf{o}}) = W\ (t_j; A_{\mathsf{o}}) + (t \quad t_j)(C \quad \rho),\ t\ \mathit{2}\ [t_j, b_j]; \tag{15}$$

$$W\ (t; A_{\mathsf{o}}) = [W\ (b_{j\ \ 1}; A_{\mathsf{o}}) \quad \rho(t \quad b_{j\ \ 1})]^+, \quad t\ \mathit{2}\ [b_{j\ \ 1}, t_j]; \tag{16}$$

for $j = 1, 2, \ldots$. Equations (12)–(16) provide a complete characterization of the virtual workloads of the traffic flows A_{i} and A_{o} and can be used to construct the corresponding workload curves shown in Fig. 3.

B. Internal Traffic Workload Analysis

To analyze the stochastic $(\sigma\ , \rho)$ regulator developed in Section IV, it will be convenient to introduce the internal traffic flow A_1 shown in Fig. 2 for the (σ, ρ) regulator and in Fig. 4 for the $(\sigma\ , \rho)$ regulator. We shall develop some new results for the (σ, ρ) regulator involving the internal flow A_1, which will be useful in the design of the $(\sigma\ , \rho)$ regulator. Figure 2 can be viewed as a more detailed depiction of the (σ, ρ) regulator shown as a single box in Fig. 1. The diagrams in Figs. 2 and 4 represent single-server, infinite buffer queueing systems. The box represents the server, which imposes a variable service delay on an arriving packet. The service delay will be zero if no shaping is needed. Only one packet can reside in the server at any given time. A new packet j can arrive to the server at the instant packet $j \quad 1$ leaves the server. Packets that arrive when the server is occupied are stored in the front-end buffer in FCFS order. The traffic flow A_1 consists of the sequence of packets arriving to the server.

Let $\tilde{s}_j$ denote the arrival time of the jth packet at the buffer and let $\tilde{a}_j$ denote the complete arrival time to the buffer, i.e., $\tilde{a}_j := \tilde{s}_j + L_j/C$. The server incurs a delay on the jth packet such that it begins departing the buffer at time t_j and completely leaves the regulator at time b_j. Since the front-end buffer delays each packet until the complete departure time of the previous packet from the regulator, we have

$$\tilde{s}_j = \max\{s_j, b_{j\ \ 1}\}. \tag{17}$$

Therefore, the operation of (σ, ρ) regulator can also be described in terms of the workload $W\ (\tilde{s}_j; A_1)$. In other words, we have the following result, which is proved in Appendix A.

Proposition 1. The departure time t_j for the jth packet in the (σ, ρ) regulator is given by (cf. (8)):

$$t_j = [W\ (\tilde{s}_j; A_1) \quad \sigma]^+/\rho + \tilde{s}_j. \tag{18}$$

An example sample path of the workloads of traffic flows A_{i}, A_1, and A_{o} for a deterministic (σ, ρ) regulator is shown in the top graph of Fig. 3. If the input traffic flow A_{i} conforms to the (σ, ρ) traffic burstiness parameter at arrival times, then the workloads of A_{i}, A_1, and A_{o} will all coincide, which occurs in the interval $[s_1, s_3]$ in the figure. Within this interval, for packets $j = 1$ and 2, we have $s_j = \tilde{s}_j = t_j$ and $a_j = \tilde{a}_j = b_j$, since both packets arrive when the workload $W\ (t; A_{\mathsf{i}}) \quad \sigma$. At time $s_3 = \tilde{s}_3$, the workloads of A_1 and A_{o} diverge because packet 3 arrives when $W\ (t; A_{\mathsf{i}}) > \sigma$. Thus, the packet is delayed in the server and $t_3 > \tilde{s}_3$. However, the workloads of A_1 and A_{o} once again coincide at time b_3, i.e., the complete departure time of packet 3 from the regulator.

The workload curves of A_1 and A_{o} form a parallelogram in the interval $[\tilde{s}_3, b_3]$. The other points of this parallelogram occur at $\tilde{a}_3$, i.e., when packet 3 completely arrives to the server and at t_3, i.e., when packet 3 starts to depart the server. Then the two workload curves coincide in the interval $[b_3, \tilde{s}_4]$. In general, the workloads of A_1 and A_{o} form a (possibly degenerate) parallelogram during the interval $[\tilde{s}_j, b_j]$ and coincide during the interval $[b_j, \tilde{s}_{j+1}]$, for $j = 1, 2, \ldots$.

In Fig. 3, we see that the workload curves of A_{i} and A_1 coincide until time s_5, which is the start time of the arrival of packet 5 to the regulator. At this time, packet 4 is at the server, so packet 5 waits until time $\tilde{s}_5 > s_5$ to go into service. At time $\tilde{a}_5$, when packet 5 has arrived completely to the server, the two curves coincide once again. In the interval $[s_5, \tilde{a}_5]$, the two curves form a parallelogram. This is not true in general, but in the interval $[s_j, \tilde{a}_j]$ a (possibly degenerate) parallelogram can be formed in which the sides consists of $W\ (t; A_{\mathsf{i}})$ for $t\ \mathit{2}\ [s_j, a_j]$, $W\ (t; A_1)$ for $t\ \mathit{2}\ [\tilde{s}_j, \tilde{a}_j]$, $W\ (t; A_{\mathsf{i}})$ for $t\ \mathit{2}\ [a_j, \tilde{a}_j]$, and $W\ (t; A_1)$ for $t\ \mathit{2}\ [s_j, \tilde{s}_j]$ for $j = 1, 2, \ldots$. Thus, the workload curves of A_{i} and A_1 are separated by a *sequence* of possibly degenerate parallelograms. Each such parallelogram corresponds to a packet delayed in the buffer of the regulator. A similar type of relationship holds between the workload curves of A_{i} and A_{o}. The workload curves of A_1 and A_{o} are separated by at most *one* parallelogram because the server can hold at most one packet.

Based on the above analysis and Proposition 1, the operation of the (σ, ρ) regulator can be characterized in terms of the internal traffic flow A_1 and the output traffic A_{o}. Analogous to equations (12)–(16) the following equations involving A_1 can be derived:

$$W\ (t; A_{\mathsf{o}}) = W\ (t; A_1) = [W\ (b_{j\ \ 1}; A_{\mathsf{o}}) \quad \rho(t \quad b_{j\ \ 1})]^+, \quad t\ \mathit{2}\ [b_{j\ \ 1}, \tilde{s}_j]; \tag{19}$$

$$W\ (t; A_1) = W\ (\tilde{s}_j; A_1) + (t \quad \tilde{s}_j)(C \quad \rho),\ t\ \mathit{2}\ [\tilde{s}_j, \tilde{a}_j]; \tag{20}$$

$$W\ (t; A_1) = W\ (\tilde{a}_j; A_1) \quad \rho(t \quad \tilde{a}_j),\ t\ \mathit{2}\ [\tilde{a}_j, b_j]; \tag{21}$$

$$W\ (t_j; A_{\mathsf{o}}) = \begin{array}{ll} \sigma, & \text{if } W\ (\tilde{s}_j; A_1) > \sigma, \\ W\ (\tilde{s}_j; A_1), & \text{if } W\ (\tilde{s}_j; A_1) \quad \sigma; \end{array} \tag{22}$$

$$W\ (t; A_{\mathsf{o}}) = W\ (t_j; A_{\mathsf{o}}) + (t \quad t_j)(C \quad \rho),\ t\ \mathit{2}\ [t_j, b_j]; \tag{23}$$

$$W\ (t; A_{\mathsf{o}}) = W\ (\tilde{s}_j; A_{\mathsf{o}}) \quad \rho(t \quad \tilde{s}_j),\ t\ \mathit{2}\ [\tilde{s}_j, t_j]; \tag{24}$$

for $j = 1, 2, \ldots$. Equation (19) follows from the following

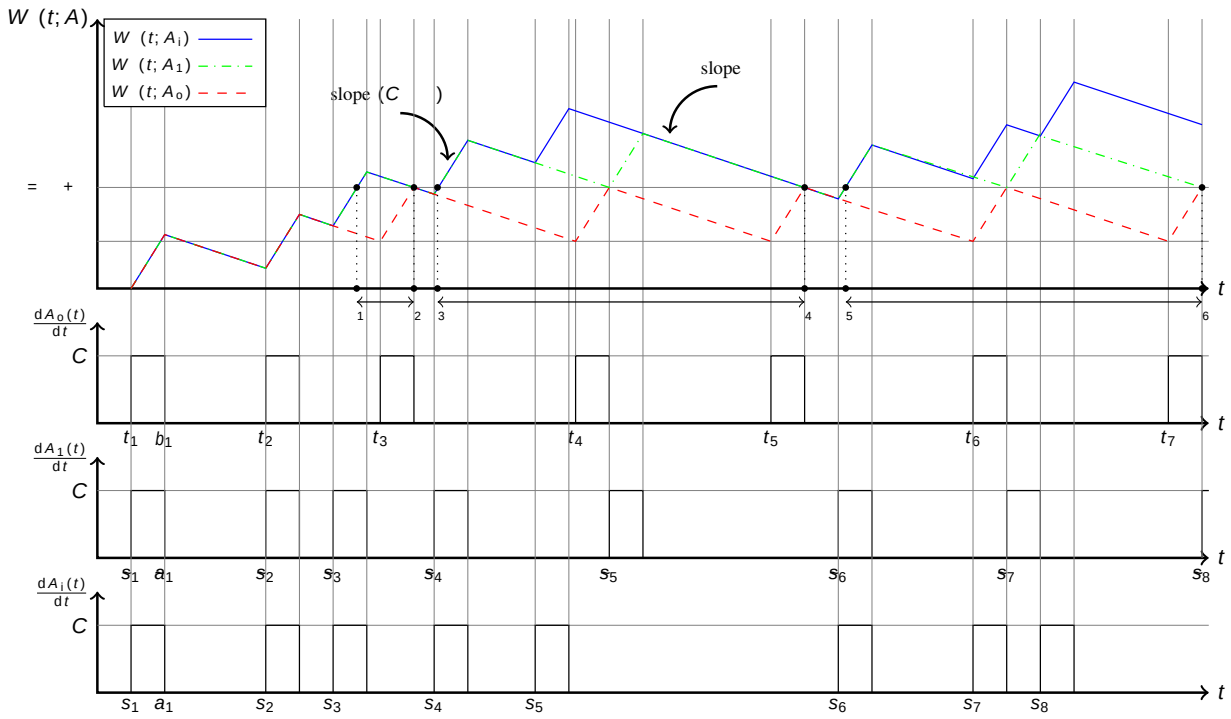


Fig. 3. Example of the operation of a (;) traffic regulator.

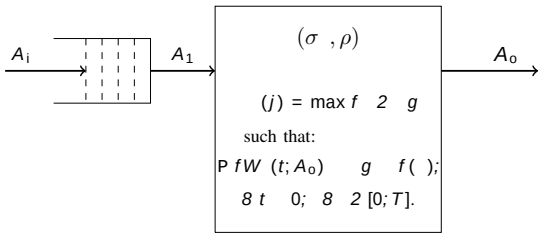


Fig. 4. Idealized stochastic (;) traffic regulator.

equality

$$W\ (b_j\ \ _1; A_o) = W\ (b_j\ \ _1; A_1), \tag{25}$$

which can be verified using (20)-(24) and (9). Intuitively, (25) holds because at most one packet is in the server of the regulator at any given time.

IV. Stochastic $(\sigma\ , \rho)$ Regulator

To our knowledge, the problem of regulating a traffic flow to force conformance to a stochastic traffic bound has not been addressed in the literature. This motivates the development of a *stochastic* traffic regulator, which enforces a probabilistic bound on a traffic source as follows:

$$\mathsf{P}\ fW\ (t; A_o)\quad \gamma g\quad f(\gamma), \quad 8t\quad 0,\ 8\gamma\ 2\ [0, T], \tag{26}$$

where f is a non-increasing positive bounding function and T is a limit on the tail distribution of the workload (see [15]). As $T\ \ !\ \ 1$, (26) becomes equivalent to the gSBB bound in (3).

A. Operational Principles

We shall show that enforcement of (26) can be achieved under steady-state conditions using a regulator with a constant rate parameter ρ and a variable burstiness parameter σ which is chosen from a finite set Σ for each arriving packet. We refer to such a regulator as a stochastic $(\sigma\ , \rho)$ regulator. A schematic of an idealized $(\sigma\ , \rho)$ regulator is shown in Fig. 4. The input and output links of the regulator are assumed to have capacity C. A buffer at the front-end of the regulator delays incoming packets until all previous packets have departed, thus ensuring a FCFS service discipline. Let A_i and A_o denote, respectively, the input traffic to and output traffic from the regulator. We denote the internal traffic departing from the front-end buffer by A_1. Let s_j and $\tilde{s}_j$ denote, respectively, the arrival and departure times of the jth packet at the buffer.

For each packet j, the $(\sigma\ , \rho)$ regulator chooses a burstiness parameter $\sigma\ (j)$ such that a delay d_j is incurred, where (cf. (8))

$$d_j = t_j\quad s_j = [W\ (s_j; A_i)\quad \sigma\ (j)]^+/\rho, \tag{27}$$

and t_j denotes the time at which the packet starts departing the traffic regulator. The packet completely leaves the regulator at time b_j. The front-end buffer acts as in the deterministic (σ, ρ) regulator (see Section III); therefore $\tilde{s}_j$ can be derived from (17). As in a deterministic (σ, ρ) traffic regulator, the rate parameter ρ should be greater than the long-term average input traffic rate, i.e.,

$$\rho > \lim_{t\ !\ 1} \frac{A(t)\quad A(s)}{t\quad s}, \quad 8s\quad 0, \tag{28}$$

to avoid incurring unbounded packet delay.

B. Overshoot Probability and Overshoot Ratio

To design a practical (σ, ρ) regulator, the *overshoot probability* $\mathsf{P}\{W(t; A_o) > \gamma\}$ in (26) can be approximated by a time-averaged *overshoot ratio*.

Definition 5. Given a threshold value $\zeta > 0$ and a traffic flow A, an *overshoot interval* with respect to A and ζ is a maximal interval of time η such that $W(\tau; A) > \zeta$ for all $\tau \in \eta$. Let $|\eta|$ denote the length of interval η. Let $\mathcal{O}(t)$ denote the set of overshoot intervals contained in $[0, t]$. Then the *overshoot duration* up to time t is defined as

$$O(t; A) = \sum_{\eta \in \mathcal{O}(t)} |\eta|. \tag{29}$$

In Fig. 3, the overshoot set with respect to threshold value ζ until the end of time domain depicted in the figure consists of three intervals $[\tau_1, \tau_2]$, $[\tau_3, \tau_4]$ and $[\tau_5, \tau_6]$. Given a time interval $[a, b]$, let $w_1 = W(a; A_o)$ and $w_2 = W(b; A_o)$. We define the increment in overshoot duration when the workload of the output process is *increasing* due to a packet departure from the regulator as follows:

$$\alpha(a, b, \zeta) = \begin{cases} b - a, & \zeta \geq w_1, \\ (w_2 - \zeta)/(C - \rho), & w_1 > \zeta \geq w_2, \\ 0, & w_2 < \zeta. \end{cases} \tag{30}$$

We define the increment in overshoot duration when the workload is *decreasing* due to the packet inter-departure time as follows:

$$\beta(a, b, \zeta) = \begin{cases} b - a, & \zeta \geq w_2, \\ (W_1 - \zeta)/\rho, & w_2 > \zeta \geq w_1, \\ 0, & w_1 < \zeta. \end{cases} \tag{31}$$

Figure 5 illustrates $\alpha(a, b, \zeta)$ and $\beta(a, b, \zeta)$. The following proposition follows immediately from the definitions and shows how to compute $O(t; A_o)$ at time $t = b_j$ for packet j.

Proposition 2.

$$O(b_1; A_o) = \alpha(t_1, b_1, \zeta),$$
$$O(b_j; A_o) = O(b_{j-1}; A_o) + \beta(b_{j-1}, t_j, \zeta) + \alpha(t_j, b_j, \zeta),$$

for $j = 2, 3, \ldots$.

We define the *overshoot ratio* of the regulator at time t with respect to a threshold ζ by

$$o(t) = O(t; A_o)/t. \tag{32}$$

For sufficiently large t, the virtual workload $W(t)$ can be modeled as a stationary and ergodic process [19]. This is a reasonable assumption when ρ satisfies (28), since the queueing system is stable in this case. Under this assumption, the overshoot ratio asymptotically approaches the overshoot probability, i.e.,

$$o(t) \to \mathsf{P}\{W(t; A_o) > \zeta\} \text{ as } t \to \infty. \tag{33}$$

Using the overshoot ratio as a proxy for the overshoot probability in (26), we design a (σ, ρ) regulator that selects the burstiness parameters $\sigma(j)$, $j = 1, 2, \ldots$, from a finite set Σ such that

$$o(t) \leq f(\gamma), \quad \forall t \in [b_{j-1}, b_j], \ \forall \gamma \in [0, T], \tag{34}$$

while minimizing the incurred packet delay. Note that the bound (34) is satisfied at any time t, whereas the approximation for the overshoot probability in (33) holds asymptotically.

C. Piecewise-Linear Bounding Function

Next, we address the issues of selecting the set Σ of burstiness parameter values and verification of the condition (34). We replace the bounding function f by a piecewise-linear function $\bar{f}$ defined in terms of a set of values $T_1 < T_2 < \ldots < T_M$ and the value δ given by (2) satisfying the following constraints:

$$T - T_{M-1} \geq \delta; \ T_M \geq T, \ T_1 \geq \delta, \ T_{i+1} - T_i \geq \delta, \tag{35}$$

for $i = 1, 2, \ldots, M - 1$. For given T and δ, the maximum possible value of M is given by

$$M_{\max} = \lfloor T/\delta \rfloor - 1. \tag{36}$$

The values $\{T_1, \ldots, T_M\}$ determine the set of burstiness parameter values

$$\Sigma = \{\sigma_i := T_i - \delta : i = 1, \ldots, M\}. \tag{37}$$

Note that $\sigma_1 < \sigma_2 < \ldots < \sigma_M$.

Without loss of generality, we assume $f(0) = 1$. The function $\bar{f}$ is designed[1] such that $f \leq \bar{f}$ in $[0, T_1)$, $\bar{f} \geq f$ in $[T_1, T]$ and $\|f - \bar{f}\|$ is small in $[T_1, T]$, where $\|\cdot\|$ denotes a norm on the space of bounding functions $\mathcal{BF}$, e.g., the L^2-norm $\|\cdot\|_2$. Since $\bar{f}$ is chosen from the class of piecewise-linear functions with a finite number M of linear pieces, it cannot be chosen arbitrarily close to f, although $\|f - \bar{f}\|$ decreases as M is increased. In particular, we set $\bar{f}(\gamma) = f(0) = 1$ for $\gamma \in [0, T_1)$. Since $\bar{f} \geq f$ in this interval, traffic regulation with respect to $\bar{f}$ may result in violation of (26). However, the violation probability is upper bounded by T_1/T, which can be made arbitrarily small by suitable choices of T_1 and/or T. We also set $\bar{f}(\gamma) = \bar{f}(T)$ for $T_M > \gamma \geq T_{M-1}$, and we choose a large value for T_M such that the burst size of the output traffic is not limited by the stochastic (σ, ρ) regulator.

In the interval $[T_i, T_{i+1})$ let

$$g_i(\gamma) = f(T_{i+1}) + \omega_i(\gamma - T_{i+1}) \tag{38}$$

represent the line connecting the points $(T_i, f(T_i))$ and $(T_{i+1}, f(T_{i+1}))$ with slope

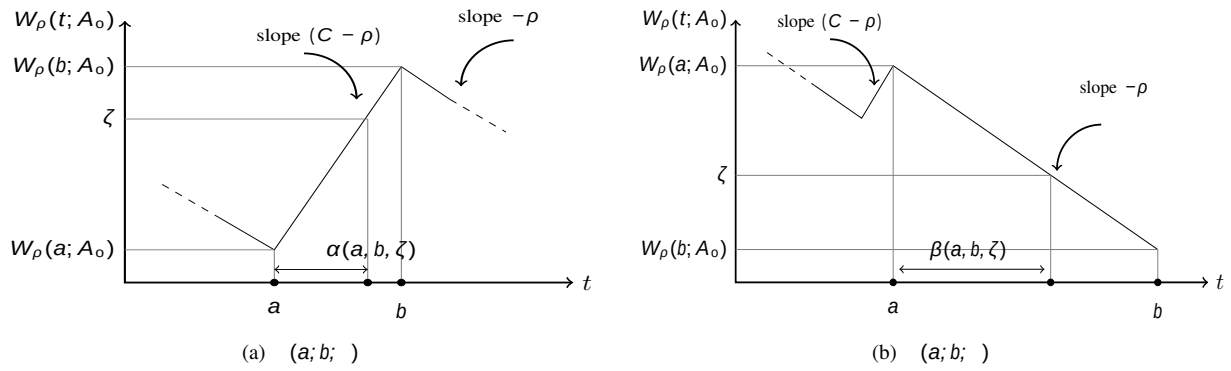
$$\omega_i = \frac{f(T_{i+1}) - f(T_i)}{T_{i+1} - T_i} \tag{39}$$

for $i = 1, \ldots, M - 2$. If $f(\gamma) \geq g_i(\gamma)$ for all $\gamma \in [T_i, T_{i+1})$ we set $\bar{f} = g_i$ in this interval. Otherwise, we set $\bar{f} = h_i$ on $[T_i, T_{i+1})$, where

$$h_i(\gamma) = f(T_{i+1}) + f^0(T_{i+1})(\gamma - T_{i+1}). \tag{40}$$

This ensures that $\bar{f} \geq f$ on $[T_1, T_{M-1})$. We then set $\bar{f}(\gamma) = f(T)$ for $\gamma \in [T_{M-1}, T_M]$ and $\bar{f}(\gamma) = 0$ for $\gamma > T_M$. To

[1]For technical reasons, a slightly different definition of f for values of $M < M_{\max}$ is used in the proofs of Theorems 1–3 given in [20].



(a) $(a;b;\ )$ (b) $(a;b;\ )$

Fig. 5. Calculating the increment in the overshoot duration.

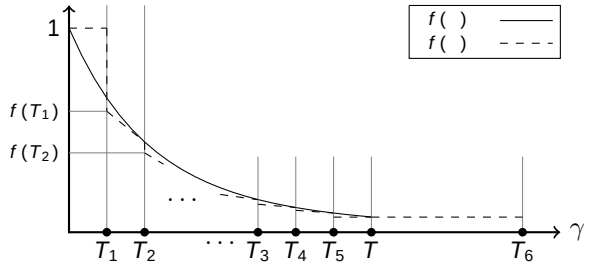


Fig. 6. Piecewise-linear approximating function f, $M = 6$.

summarize, we define

$$\bar{f}(\gamma) = \begin{cases} 1, & \gamma \in [0, T_1), \\ f(T_{i+1}) + m_i(\gamma - T_{i+1}), & \gamma \in [T_i, T_{i+1}), \\ f(T), & \gamma \in [T_{M-1}, T_M], \\ 0, & \gamma > T_M, \end{cases} \tag{41}$$

for $i = 1, \ldots, M-2$ and the slopes m_i are given by

$$m_i = \begin{cases} \omega_i, & \text{if } f \neq g_i \text{ on } [T_i, T_{i+1}), \\ f^0(T_{i+1}), & \text{otherwise,} \end{cases} \tag{42}$$

for $i = 1, \ldots, M-2$.

D. Canonical (σ, ρ) Regulator

Based on the definition of $\bar{f}$ in (41), we modify the constraint in (34) to hold only for $\gamma \in [T_1, T]$, i.e.,

$$o(t) \leq f(\gamma), \quad \forall t \in [b_{j-1}, b_j], \ \forall \gamma \in [T_1, T]. \tag{43}$$

Towards a practical implementation, we further replace the bounding function f by $\bar{f}$ to obtain the following burstiness constraint:

$$o(t) \leq \bar{f}(\gamma), \quad \forall t \in [b_{j-1}, b_j], \ \forall \gamma \in [T_1, T]. \tag{44}$$

To incur minimal packet delay, $\sigma(j)$ should be chosen as the largest value in Σ such that the constraint (44) is maintained. We then define a *canonical* (σ, ρ) regulator as follows:

$$A_j = \{\sigma \in \Sigma : o(t) \leq \bar{f}(\gamma), \forall t \in [b_{j-1}, b_j(\sigma)], \forall \gamma \in [T_1, T]\}$$
$$\sigma(j) = \begin{cases} \sigma_{\max A_j}, & \text{if } A_j \neq \emptyset, \\ \sigma_1, & \text{otherwise.} \end{cases} \tag{45}$$

Equations (18)-(24) are the governing equations for a stochastic (σ, ρ) in which σ is replaced by $\sigma(j)$ according to (45). The canonical regulator cannot be implemented directly, since the condition for A_j in (45) is impractical to verify for all values of $t \in [b_{j-1}, b_j]$ and $\gamma \in [T_1, T]$. Next, we develop practical implementations of the canonical (σ, ρ) regulator.

E. Basic Implementation

We assume that T_M is chosen sufficiently large such that for every packet j the set

$$B_j = \{1 \leq \ell \leq M : \sigma_\ell \geq W(\tilde{s}_j; A_1)\}, \tag{46}$$

is non-empty. Let

$$I_j = \{2 \leq \ell \leq \min B_j : o_{T_{\ell-1}}(b_j(\sigma_\ell)) \leq \bar{f}(T_{\ell-1})\} \tag{47}$$

where $t_j(\sigma_\ell)$ and $b_j(\sigma_\ell)$ are given by (18) and (9), respectively. Let

$$\sigma(j) = \begin{cases} \sigma_{\max I_j}, & \text{if } I_j \neq \emptyset, \\ \sigma_1, & \text{otherwise.} \end{cases} \tag{48}$$

Equations (46)–(48) are used to develop approximate implementations of the canonical (σ, ρ) regulator given by (45). For a given value of $\sigma_\ell \in \Sigma$, the condition in (47) is checked only at $t = b_j(\sigma_\ell)$ and $\gamma = T_{\ell-1}$. Therefore, as shown in Section V, the constraint (43) may be violated for some values of t. However, these violations will not occur when t is sufficiently large.

Theorem 1. The (σ, ρ) regulator defined by (46)–(48) produces output traffic that satisfies (43) for sufficiently large t.

Thus, the proposed regulator ensures that the overshoot ratio $o(t)$ of the output traffic is bounded by the function $f(\gamma)$ for sufficiently large t. Since, by (33), $o(t) \leq \mathsf{P}\{fW(t) > \gamma g\}$, we have that $\mathsf{P}\{fW(t) > \gamma g\} \leq f(\gamma)$ for sufficiently large t i.e., the gSBB bound (3) is satisfied by the output traffic. The proof of Theorem 1 is somewhat involved and can be found in [20]. A pseudo-code implementation of the stochastic (σ, ρ) regulator is given in Algorithm 1. The input traffic A_i is represented as a sequence $\{(s_1, L_1), \ldots, (s_N, L_N)\}$, where the s_i's are the arrival times of the packets and the L_i's are the packet lengths. The (σ, ρ) regulator consists of the rate ρ, the bounding function f, the range T over which the bound is applied, the set Σ, and the values $\{T_1, \ldots, T_M\}$,

Algorithm 1 $(\sigma\ ,\rho)$ stochastic regulator

Input: A_{i} $f(s_1,L_1),\ldots,(s_N,L_N)g$; ▷ Input traffic
Input: ρ; $f(\ )$; T; M; L_{max}; C ▷ Regulator parameters
Output: A_{o} $ft_1,t_2,\ldots,t_Ng$ ▷ Output traffic
1: δ $(1$ $\rho/C)L_{\mathsf{max}}$
2: Compute T_i,σ_i for $i=1,2,\ldots,M$ ▷ (35), (37)
3: Compute $\bar{f}(\ )$ ▷ (41)
4: t_1 $\tilde{s}_1$ s_1; b_1 t_1+L_1/C
5: $W\ (\tilde{s}_1;A_1)$ $W\ (t_1;A_{\mathsf{o}})$ 0
6: Compute $W\ (b_1;A_{\mathsf{o}})$ ▷ (23)
7: Compute $o_{T_i}(b_1)$; $i=1,2,\ldots,M$ 1 ▷ Prop. 2
8: **for** $j=1,\ldots,N$ **do** ▷ Packet j arrives at time s_j
9: Compute $\tilde{s}_j$, $W\ (\tilde{s}_j;A_1)$, B_j ▷ (17), (19), (46)
10: found **false**; k $\min B_j$
11: **for** $\ell=k,\ldots,2$ **do** ▷ k 2
12: σ σ_{ℓ}; Compute $t_j(\sigma)$, $b_j(\sigma)$ ▷ (18), (9)
13: Compute $W\ (t_j;A_{\mathsf{o}})$, $W\ (b_j;A_{\mathsf{o}})$ ▷ (22), (23)
14: Compute $o_{T_{\ell\ 1}}(b_j)$ ▷ Prop. 2
15: **if** $o_{T_{\ell\ 1}}(b_j)$ $\bar{f}(T_{\ell})$ **then** ▷ (47)
16: found **true**; **break**
17: **end if**
18: **end for**
19: **if not** found **then**
20: σ σ_1; Compute $t_j(\sigma)$, $b_j(\sigma)$ ▷ (18), (9)
21: **end if**
22: Compute $o_{T_i}(b_j)$; $i=1,2,\ldots,M$ 1 ▷ Prop. 2
23: **end for**

which determine the piecewise-linear bounding function $\bar{f}$. The input and output links for the regulator are assumed to be of capacity $C>\rho$. The output traffic A_{o} is represented by the sequence $f(t_1,L_1),\ldots,(t_N,L_N)g$, where the t_i's are packet departure times. The **for** loop starting in line 11 finds the largest ℓ 2 $f2,\ldots,k=\min B_jg$ such that the inequality in (47) is satisfied with $\sigma=\sigma_{\ell}$. If such σ_{ℓ} exists, then $\sigma\ (j)=\sigma_{\ell}$; otherwise, $\sigma\ (j)=\sigma_1$, in accordance with (48).

Computation of the departure time, t_j, of the jth packet requires updates to $o_{T_i}(b_j)$ for $i=1,\ldots,M$ 1. Once t_j is determined, the values of $o_{T_i}(b_j)$, for $i=1,\ldots,M$ 1, need to be updated. Thus, the overall computational complexity is $O(M)$ per packet. Since the updates to $o_{T_i}(b_j)$ are independent of each other, they could be executed in parallel using a hardware accelerator such as a graphics processing unit (GPU). In particular, a parallel implementation of the **for** loop at line 11, can effectively reduce the computational complexity per packet to constant time, i.e., $O(1)$.

F. Alternative Implementation

The requirement of sufficiently large t in Theorem 1 can be avoided by modifying the definition of I_j in (47) to include additional checks. Let B_j be as defined in (46). We re-define I_j as follows:

$$I_j = 2\quad \ell\quad \min B_j : o_{T_i}(b_j(\sigma_{\ell}))\quad \bar{f}(T_i)\quad \epsilon_{i;j}(\sigma_{\ell}),\ 8i=1,\ldots,\ell\quad 1\ , \tag{49}$$

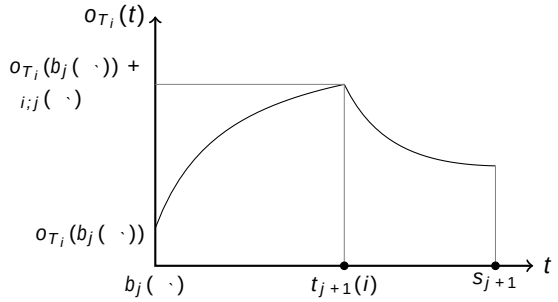


Fig. 7. Overshoot ratio $o_{T_i}(t)$ for $t>b_j$, when $W\ (s_{j+1};A_{\mathsf{o}})=0$.

where

$$\epsilon_{i;j}(\sigma_{\ell}):=\begin{cases}\dfrac{W\ (b_j(\ \ \cdot);A_{\mathsf{o}})\quad T_i}{b_j(\ \ \cdot)}(1\quad \bar{f}(T_i)), & i=1,\ldots,\ell\quad 2,\\ \bar{f}(T_{\ell\ 1})\quad \bar{f}(T_{\ell}), & i=\ell\quad 1.\end{cases} \tag{50}$$

The modified definition of I_j in (49) involves additional checks for the jth packet, which may result in a smaller value of $\sigma\ (j)$ and hence higher delay incurred on the packet. Interestingly, our numerical simulations show that this results in slightly smaller *average* delay incurred on the input traffic. This can be explained as follows. By incurring more delay on *some* input packets at an earlier stage, the output traffic may be better shaped to the desired bound; therefore, on average, less delay will need to be incurred on future packets.

The overshoot ratio $o_{T_i}(t)$ at $t=b_j(\sigma_{\ell})$ is checked against $\bar{f}(T_i)$ $\epsilon_{i;j}(\sigma_{\ell})$ rather than $\bar{f}(T_i)$, for $i=1,\ldots,\ell$ 2. The reasoning behind this stricter condition is illustrated in Fig. 7. In choosing $\sigma\ (j)=\sigma_{\ell}$, the overshoot ratios $o_{T_i}(t)$, for $i=1,\ldots,\ell$ 2, will be increasing functions of t, as shown in Fig. 7, up to time $t=t_{j+1}(i)$, which is defined as the time at which

$$W\ (t;A_{\mathsf{o}})=T_i\ \text{ for }\ i=1,2,\ldots,T_{\ell\ 2}, \tag{51}$$

and the $(j+1)$th packet arrives sufficiently late that $W\ (s_{j+1};A_{\mathsf{o}})=0$. Enforcing the condition in (49) with the lower values $\bar{f}(T_i)$ $\epsilon_{i;j}(\sigma_{\ell})$ ensures that the overshoot ratio stays less than $\bar{f}(T_i)$ for all t b_j. In this implementation, for a given value of σ_{ℓ} 2 Σ, the condition (43) is checked only at $t=b_j(\sigma_{\ell})$ and for γ 2 $fT_1,\ldots,T_{\ell\ 1}g$. These extra checks compared to Algorithm 1, as stated in the following theorem and shown in Section V, guarantee that there will be no violation of the constraint (43).

Theorem 2. The $(\sigma\ ,\rho)$ regulator defined by (46), (49), and (48) produces output traffic that satisfies (43) for all t 0.

Algorithm 2 Replacement for lines 14–17 of Algorithm 1

14: Compute $o_{T_i}(b_j(\sigma_{\ell}))$; $i=1,\ldots,\ell$ 1 ▷ Prop. 2
15: Compute $\epsilon_{i;j}(\sigma_{\ell})$; $i=1,\ldots,\ell$ 1 ▷ (50)
16: **if** $o_{T_i}(b_j)$ $\bar{f}(T_i)$ $\epsilon_{i;j}(\sigma_{\ell})$ $8i$ 2 $f1,\ldots,\ell$ $1g$ **then**
17: found **true**; **break**
18: **end if**

A proof of Theorem 2 is given in Appendix B. By modifying

Algorithm 1 in accordance with Theorem 2, we obtain an alternative implementation that satisfies (43) for all $t \geq 0$ at the expense of some additional computation. The modified implementation is obtained by replacing lines 15–18 in Algorithm 1 with the pseudo-code shown in Algorithm 2. In lines 15 and 16, $\ell - 1$ values of $o_{T_i}(b_j(\sigma_\cdot))$ and $\epsilon_{i;j}(\sigma_\cdot)$ need to be computed. Therefore, the complexity of the **for** loop at line 12 in Algorithm 1 is $O(M^2)$ and the overall complexity of the modified algorithm is $O(M^2)$ per packet.

With further algorithmic modifications, the complexity of Algorithm 2 can be reduced to $O(M)$, i.e., the same time complexity as Algorithm 1. Let $\mathcal{B}_j$ be as in (46). Let $k = \min \mathcal{B}_j$ and

$$\mathcal{J}_j = \{1 \leq \ell \leq k-1 : o_{T_\cdot}(b_j(\sigma_k)) \leq \bar{f}(T_\cdot) - \epsilon_{\cdot;j}(\sigma_k)\}, \quad (52)$$

where $\epsilon_{i;j}(\sigma_k)$ is defined in (50). If $1 \in \mathcal{J}_j$ let

$$m = \max\{\ell \in \mathcal{J}_j : i \in \mathcal{J}_j, \ \forall 1 \leq i \leq \ell\}, \quad (53)$$

and let

$$\mathcal{K}_j = \{2 \leq \ell \leq m+1 : o_{T_{\cdot\ 1}}(b_j(\sigma_\cdot)) \leq \bar{f}(T_\cdot)\}, \quad (54)$$

where $b_j(\sigma_\cdot)$ and $o_{T_{\cdot\ 1}}(b_j(\sigma_\cdot))$ are given as follows:

$$b_j(\sigma_\cdot) = \tilde{s}_j + (W(\tilde{s}_j; A_1) - \sigma_\cdot)/\rho + L_j/C, \quad (55)$$

$$b_j(\sigma_\cdot)o_{T_{\cdot\ 1}}(b_j(\sigma_\cdot)) = b_j(\sigma_k)o_{T_{\cdot\ 1}}(b_j(\sigma_k)) + (W(\tilde{s}_j; A_1) - \sigma_\cdot)/\rho. \quad (56)$$

We now present a third implementation of the canonical (σ, ρ) regulator given by

$$\sigma(j) = \begin{cases} \sigma_{\max \mathcal{K}_j}, & \text{if } 1 \in \mathcal{J}_j \text{ and } \mathcal{K}_j \neq \emptyset, \\ \sigma_1, & \text{otherwise.} \end{cases} \quad (57)$$

Theorem 3. The (σ, ρ) regulator defined by (46) and (52)–(57) produces the same output traffic as the (σ, ρ) regulator of Theorem 2 for a given input flow and hence the output flow satisfies (43) for all $t \geq 0$.

Algorithm 3 Replacement for lines 11–18 of Algorithm 1

```
11: m ← 0; found ← false
12: for ℓ = 1, ..., k − 1 do                          ▷ k − 2
13:     σ_ℓ ← σ_k; Compute t_j(σ), b_j(σ),            ▷ (18), (9)
14:     Compute ε_{·;j}(σ), o_{T·}(b_j)               ▷ (50), Prop. 2
15:     if o_{T·}(b_j) > f̄(T·) − ε_{·;j}(σ) then      ▷ (52)
16:         break
17:     end if
18:     m ← m + 1
19: end for
20: for ℓ = m + 1, ..., 2 do
21:     σ_ℓ ← σ·; Compute b_j(σ), o_{T· 1}(b_j)       ▷ (55), (56)
22:     if o_{T· 1}(b_j) ≤ f̄(T·) then                 ▷ (49)
23:         found ← true; break
24:     end if
25: end for
```

A proof of Theorem 3 is given in Appendix C. The (σ, ρ) regulator corresponding to Theorem 3 can be implemented by replacing lines 11-18 in Algorithm 1 with the lines shown in Algorithm 3. The **for** loops at lines 11 and 19 in Algorithm 3 both have complexity $O(M)$. Therefore, the overall complexity of Algorithm 3 is $O(M)$ per packet. As with Algorithm 1, using a suitable parallel implementation, the complexity per packet can be further reduced to $O(1)$.

To summarize, Algorithm 3 provides a theoretical guarantee that the sample path bound (43) holds for all $t \geq 0$. Algorithm 1 provides the weaker guarantee that the sample path bound is satisfied asymptotically. However, our empirical studies have shown that Algorithm 1 achieves the bound (43) very quickly in practice. Both implementations have computational complexity $O(M)$. Our simulation results (see Tables I and II in Section V), show that Algorithm 1 incurs higher average delay and greater variance in shaping the traffic to the desired gSBB bound.

V. Numerical Results

We evaluate the performance of the (σ, ρ) regulator first in a basic scenario with Poisson-like traffic and then a more realistic example with bursty traffic.

A. Basic Scenario

First, we consider a system similar to one studied in [3]. The packets sizes L_j are drawn randomly according to

$$L_j \sim \mathrm{U}\{L_{\min}, L_{\min}+1, \ldots, L_{\max}\}, \quad (58)$$

where $\mathrm{U}(\mathcal{A})$ denotes a uniform distribution over the set $\mathcal{A}$. The inter-arrival times of the packets, $s_{j+1} - s_j$, are determined as follows:

$$s_{j+1} - s_j = U_j + L_j/C, \quad (59)$$

where $U_j \sim \mathrm{Exp}(\lambda)$, i.e., $\{U_j\}$ is an i.i.d. sequence of exponentially distributed random variables with rate parameter λ. By adopting (59) to model the inter-arrival times, we ensure that packets are received after the previous ones have been fully received, i.e., the packets will not overlap with each other. In a system described by (58)–(59), $\rho^{-1}W(s_j; A_i)$ is equal to the waiting time experienced by the jth customer in a $G/G/1$ queueing system in which the service time of the jth customer is given by $S_j = (\rho^{-1} - C^{-1})L_j$ and the inter-arrival time between the jth and $(j+1)$th customer is U_j [3], [16], [21].

In this example, we set $L_{\min} = 5$, $L_{\max} = 10$, and $\lambda = 0.25$, and $\rho = 0.65$. We use the following bounding function:

$$f(\sigma) := \begin{cases} 2.5 - 10^{-3}\sigma + 1, & 0 \leq \sigma \leq 40, \\ 5 - 10^{-3}\sigma + 1.1, & 40 < \sigma \leq T = 200. \end{cases} \quad (60)$$

In Fig. 8, $\bar{f}$ is defined by approximating f by a piecewise-linear function according to (41) with $M = 20$, $T_M = 400$ and $T_{i+1} - T_i = 20$ for $i = 1, \ldots, M-2$. Note that, as $f(\gamma)$ is also piecewise-linear, $\bar{f}(\gamma) = f(\gamma)$ for $\gamma \in [T_1, T]$. Observe that the output traffic is shaped to satisfy the desired bound.

Using the same model for inter-arrival and packet lengths, we have investigated the impact of the parameter M on traffic shaping of the input traffic. From Fig. 9, we see that as M is increased, a closer fit of the output traffic to the desired bound can be achieved. In our example, the maximum possible value of M, given by (36), is $M_{\max} = 56$, for which a very close fit to the bound is achieved. Figures 8 and 9 were

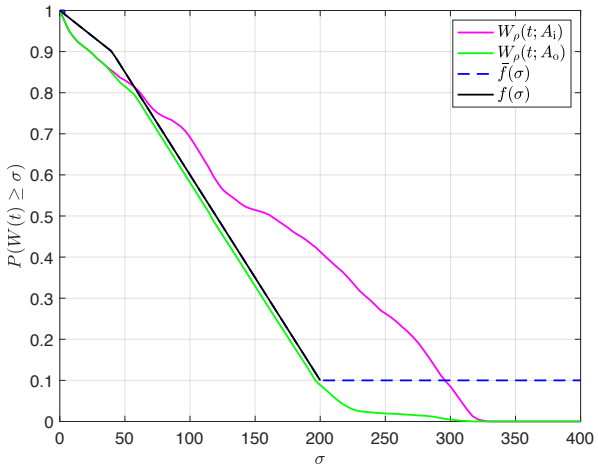


Fig. 8. Performance of the stochastic (;) traffic regulator.

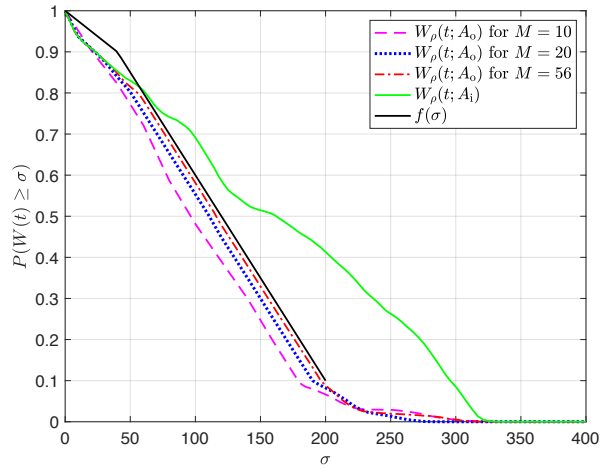


Fig. 9. Traffic regulator performance with different M values.

obtained using Algorithm 3. In Fig. 8, the gSBB bound given by (60) is represented by the black curve. The pink curve, representing the input traffic, violates this bound. The green curve, which corresponds to the output traffic, shows that the $(\sigma\ ,\rho)$ regulator succeeds in enforcing the gSBB bound.

Table I presents the average delay and standard deviation of the delay for the packets using Algorithms 1 and 3. Note that as M increases the average delay decreases and the standard deviation of the packet delay also decreases. An increase in M implies that the delay incurred on a packet can increase in smaller increments, resulting in smaller overall variance. A larger value of M results in a smaller average delay since in this case the piecewise-linear function $\bar{f}$ given by Fig. 9 also shows the impact of increasing the value of M. With larger values of M, the workload tail probability of the output is closer to the gSBB bound, i.e., the bound becomes less conservative, since f is better approximated by $\bar{f}$.

Algorithm 3 slightly outperforms Algorithm 1 for larger values of M with respect to mean and standard deviation of shaping delay, in particular, $M = 56$, as shown in Table I. In Fig. 10, a sample path of the overshoot ratio $o_{T_{17}}(t)$ is

TABLE I
TRAFFIC SHAPING DELAY WITH DIFFERENT M VALUES FOR ALGORITHM 1 AND ALGORITHM 3.

	Average Delay		Std. Dev. of Delay	
M	Alg. 1	Alg. 3	Alg. 1	Alg. 3
10	89	89	115	115
20	78	78	109	109
56	72	71	100	99

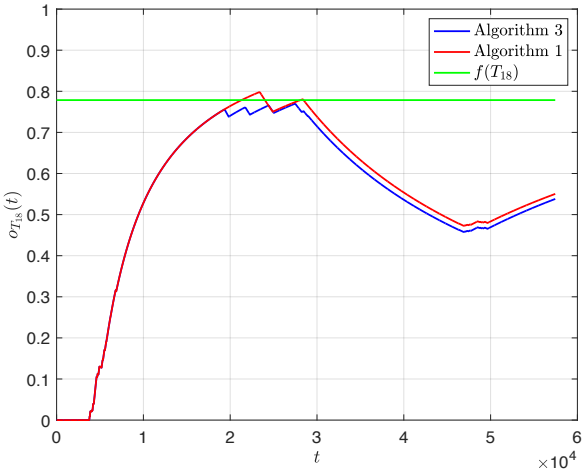


Fig. 10. Overshoot ratio $o_{T_{17}}(t)$ for $M = 56$.

shown for Algorithms 1 and 3. Observe that some violations of (34) occur with Algorithm 1 but there are no violations with Algorithm 3, which confirms Theorem 2.

B. Bursty Traffic Scenario

Next, we consider a more realistic scenario with bursty traffic. The purpose of this example is to show how the $(\sigma\ ,\rho)$ regulator can be applied to guarantee a delay bound for a traffic flow at a multiplexer and contrast this with an equivalent delay bound that can be provided using a (σ,ρ) regulator. This scenario can be generalized to a multi-hop network providing end-to-end stochastic delay guarantees by applying results from stochastic network calculus [7], [9], [13].

The packet inter-arrival times are given by (59), where the sequence fU_jg is generated according to the inter-arrival times of a three-state Markov modulated Poisson Process (MMPP) with arrival matrix and rate matrix R [16], [22]. In our example, the parameters of MMPP process are chosen according to [23, p. 79], with arrival matrix

$$= \text{diag}f116, 274, 931g \tag{61}$$

in units of packets/s and rate matrix

$$\mathsf{R} = \begin{matrix} 2 \\ 4 \\ \end{matrix} \begin{matrix} 0.12594 & 0.12594 & 0 \\ 0.25 & 2.22 & 1.975 \\ 0 & 2 & 2 \end{matrix} \begin{matrix} 3 \\ \\ \end{matrix} \tag{62}$$

in units of s ^{1}. These values were derived from matching arrival process of the I, P and B frames of an MPEG-4 encoded video to the three states of the MMPP. For the given MMPP, the average arrival rate is 358 packets/s.

The packet sizes L_j are generated according to the phase-type distribution referred to as G3 in [23, Table 1] as follows:

$$L_j \quad 0.54\ \mathrm{Er}(5, 26) + 0.46\ \mathrm{Er}(5, 956), \tag{63}$$

in units of bytes, where $\mathrm{Er}(r, 1/\mu)$ denotes an r stage Erlang distribution with mean $1/\mu$ [16], [17]. This particular phase-type distribution is a mixture of Erlang distributions, which closely approximates the empirical distribution of measured Internet packet sizes obtained in [24]. We truncate the phase-type distribution at 1500 bytes, which is the MTU (Maximum Transmission Unit) for Ethernet. In addition, since the packet lengths are integer values, the random values generated according to the truncated phase-type distribution are quantized. The average packet size according to (63) is 454 bytes. In our case, however, due to truncation and quantization, the average packet size is 438 bytes. We have set the input link capacity to 10 Mbps. Because the packet inter-arrival times are given by (59), with this choice of C, the average packet arrival rate is 319 packets/s. The average bit rate of the traffic is 1.06 Mbps.

We consider a scenario in which the available bandwidth at a multiplexer is $C_o = 2$ Mbps. Since C_o exceeds the average packet rate of the traffic source, the flow can be supported. We consider two traffic descriptors:

1) phase-type descriptor $[\rho; (a, \quad, \mathbf{Q}, T)]$ (Definition 4);
2) (σ, ρ) descriptor (Definition 1).

For both traffic descriptors we set $\rho = C_o = 2$ Mbps. The black curve in Fig. 11 shows the phase-type bound obtained using [18, Algorithm 1] with a 10-component hyperexponential distribution. The parameter T is set as the maximum value of $W\ (t; A_\mathrm{i})$, i.e., $T = 75$ KBytes. This ensures that the bounding function f corresponding to the phase-type descriptor bounds $\mathsf{P}fW\ (t; A_\mathrm{i}) \quad \sigma g$ for all values of $\sigma > 0$. For the (σ, ρ) descriptor, we set $\sigma = T = 75$ KBytes to ensure that no shaping delay will be incurred by the traffic regulator.

Along with the traffic descriptor, the user specifies a maximum delay bound requirement at the multiplexer. An admission controller determines whether or not the delay requirement can be satisfied with the available bandwidth C_o. If the traffic flow is admitted, a traffic regulator is applied to ensure conformance of the traffic flow to the traffic descriptor provided by the user. In case 1), a $(\sigma\ , \rho)$ regulator is determined by finding a piecewise-linear approximation $\bar{f}$ to the bounding function f associated with the phase-type traffic descriptor (see Section IV). Fig. 11 shows the bounding function f and its approximation $\bar{f}$ (dashed blue curve). Then Algorithm 1 or 3 can be applied to enforce $\bar{f}$. Because the bounding function $\bar{f}$ is a close upper bound to $W\ (t, A_\mathrm{i})$, the workload curves at the input and output of the regulator are very close to each other. Hence, in Fig. 11, just $W\ (t, A_\mathrm{o})$ is shown. Consequently, the $(\sigma\ , \rho)$ regulator incurs negligible shaping delay.

We next consider the delay bound that can be guaranteed at the multiplexer for the two cases. Let $Q(t)$ denote the queue size at the multiplexer with constant service rate C_o and let $D(t)$ denote the delay at the multiplexer experienced by a bit arriving at time t. Note that if $C_o \quad \rho$,

$$Q(t) = W_{C_o}(t; A_\mathrm{i}) \quad W\ (t; A_\mathrm{i}). \tag{64}$$

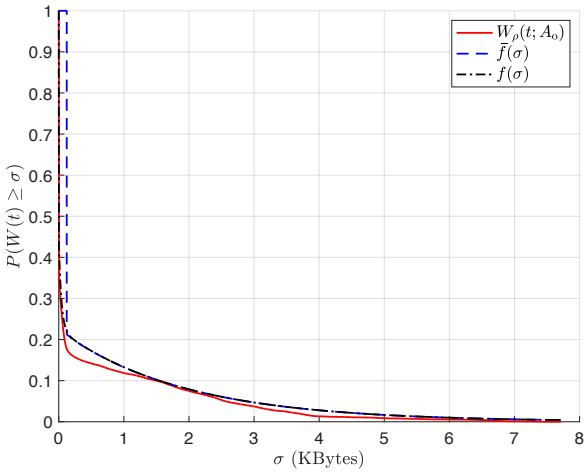


Fig. 11. $\mathsf{P}fW\ (t; A_\mathrm{o}) \quad g, f(\ )$ and $f(\ )$ vs. for bursty traffic source.

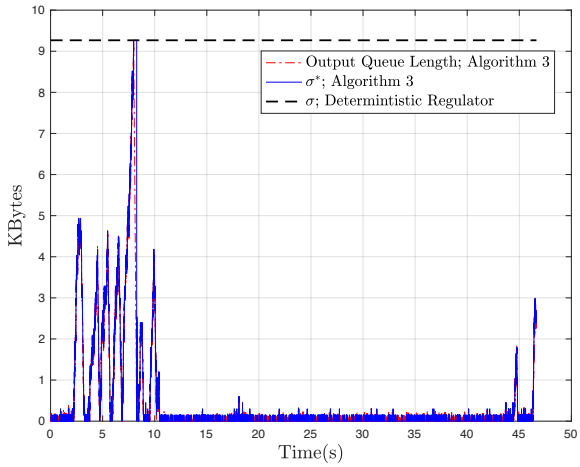


Fig. 12. Workload profile of bursty traffic fed to server with service rate and comparison of fixed bound vs. dynamic bound .

Since $\rho = C_o$, (64) holds with equality. For a FCFS server, $D(t) = Q(t)/C_o$. For the $(\sigma\ , \rho)$ regulator,

$$\mathsf{P}fD(t) \quad dg = \mathsf{P}fQ(t) \quad d\ C_o g$$
$$\mathsf{P}fW\ (t; A_\mathrm{i}) \quad d\ C_o g = \bar{f}(d\ C_o) = \epsilon. \tag{65}$$

Setting $\epsilon = 0.02$ and using the bounding curve $\bar{f}$ from Fig. 11, we find that the smallest value of d that can satisfy (65) is 22 ms. For the (σ, ρ) regulator, the smallest delay bound that can be guaranteed is $d = \sigma/C_o = 294$ ms. Thus, with the $(\sigma\ , \rho)$ regulator, a much smaller delay guarantee can be provided compared to that for the (σ, ρ) regulator.

In Fig. 12, the performance of the (σ, ρ) regulator is compared with that of a $(\sigma\ , \rho)$ regulator with $M = 63$ for a sample path of the bursty traffic source. The output queue length attains the bound σ on the burst size, but is far smaller than σ most of the time. By contrast, the value of σ closely tracks the output queue length. Clearly, the (σ, ρ) parameter provides an overly conservative bound on the traffic.

In Table II, Algorithms 1 and 3 are compared in terms of

TABLE II
TRAFFIC SHAPING DELAY INCURRED BY ALGORITHMS 1 AND 3 FOR BURSTY TRAFFIC ($M = 63$).

	Alg. 1	**Alg. 3**
Average Delay [ms]	27	1.2
Std. Dev. of Delay [ms]	50	15
$\mathbb{P}\{$shaping delay $> 0\}$	0.35	0.008

the shaping delay incurred on the bursty traffic. Algorithm 3 shapes the input traffic to the desired gSBB bound while inducing significantly less traffic shaping delay. In particular, the value of $\mathbb{P}\{$shaping delay $> 0\}$ shows that the stochastic guarantee in (65) is achieved by Algorithm 3.

VI. CONCLUSION

The stochastic traffic regulator developed in this paper addresses an open problem in the application of stochastic network calculus to real networks: enforcement of stochastic traffic bounds. Given an input traffic source, our proposed stochastic $(\sigma\ ,\rho)$ regulator inserts delays, as necessary, to ensure that the output traffic conforms to the gSBB traffic bound [9]. Operationally, the $(\sigma\ ,\rho)$ regulator works similarly to a deterministic (σ,ρ) regulator, except that the burstiness parameter σ is chosen, for each arriving packet, from among a finite set of burst size parameters, $\Sigma = \{\sigma_1,\ldots,\sigma_M\}$. We showed, through both analysis and simulation, that the $(\sigma\ ,\rho)$ regulator ensures conformance of a traffic source to a given gSBB bound. Such a bound would be negotiated between the user and the network during the admission control phase and potentially renegotiated during the lifetime of the flow (cf. [5]). A closer fit to the gSBB bound can be achieved by increasing the value of M.

APPENDIX A
PROOF OF PROPOSITION 1

Lemma A.1.

$$W\ (\tilde{s}_j; A_1) = W\ (s_j; A_i) \quad (\tilde{s}_j \quad s_j)\rho \tag{A.1}$$

A proof of Lemma A.1 is given in [20].

Proof. First, suppose $W\ (\tilde{s}_j; A_1) \quad \sigma$. Since $W\ (\tilde{s}_j; A_1) = W\ (\tilde{s}_j; A_o)$ (see (22)), we have $W\ (\tilde{s}_j; A_o) \quad \sigma$, i.e., the (σ,ρ) constraint is satisfied by the output process at time $\tilde{s}_j$. This implies that the jth packet departs the regulator starting at time $t_j = \tilde{s}_j$, which confirms (18) in this case. Next, suppose $W\ (\tilde{s}_j; A_1) > \sigma$. Then $W\ (s_j; A_i) \quad W\ (s_j; A_1) \quad W\ (\tilde{s}_j; A_1) > \sigma$. Thus, we can remove the $[\,]^+$ operator in both (8) and (18). Applying Lemma A.1 to the right-hand side of (18), we have

$$\begin{aligned}
&[W\ (\tilde{s}_j; A_1) \quad \sigma]/\rho + \tilde{s}_j \\
&= [W\ (s_j; A_i) \quad (\tilde{s}_j \quad s_j)\rho \quad \sigma]/\rho + \tilde{s}_j \\
&= [W\ (s_j; A_i) \quad \sigma]/\rho + s_j = t_j.
\end{aligned}$$

This completes the proof of Proposition 1. □

APPENDIX B
PROOF OF THEOREM 2

Here, we assume that M is set to its maximum value given by (36) and $\sigma\ (j)$ is computed using Algorithm 2. The general case, including smaller values of M, is addressed in [20].

Lemma B.1.

$$o_{T_i}(t) \quad \bar{f}(T_i), \quad \forall t \in [b_{j\ \ 1}, b_j(\sigma\ (j))], \tag{B.1}$$

for $\forall 1 \quad i \quad M$

Lemma B.2. Suppose

$$o_{T_m}(t) \quad \bar{f}(T_m) \text{ and } o_{T_{m+1}}(t) \quad \bar{f}(T_{m+1}), \tag{B.2}$$

for some $m \in \{1,2,\ldots,M \quad 2\}$ and some $t \in [b_{j\ \ 1}, b_j(\sigma`)]$. Then

$$o\ (t) \quad \bar{f}(T_m) \quad \frac{(\gamma \quad T_m)(\bar{f}(T_m) \quad \bar{f}(T_{m+1}))}{T_{m+1} \quad T_m}, \tag{B.3}$$

for all $\gamma \in [T_m, T_{m+1}]$.

Proofs of Lemmas B.1 and B.2 are given in [20].

Proof. Lemmas B.1 and B.2 imply that

$$o\ (t) < f(\gamma), \quad \forall t \in [b_{j\ \ 1}, b_j(\sigma\ (j))], \quad \forall \gamma \in [0,T], \tag{B.4}$$

if for all $i \in \{1,2,\ldots,M \quad 1\}$ and all $\gamma \in [T_i, T_{i+1}]$,

$$f(\gamma) \quad \bar{f}(T_i) \quad \frac{(\gamma \quad T_i)(\bar{f}(T_i) \quad \bar{f}(T_{i+1}))}{T_{i+1} \quad T_i}. \tag{B.5}$$

But (B.5) is true from the definition of $\bar{f}$ in (41). □

APPENDIX C
PROOF OF THEOREM 3

Lemma C.1. If $m \in J_j$ and $m < k \quad 1$ then

$$o_{T_m}(b_j(\sigma`)) + \frac{W\ (b_j(\sigma`); A_o) \quad T_m}{\rho b_j(\sigma`)}(1 \quad \bar{f}(T_m)) \quad \bar{f}(T_m), \tag{C.1}$$

for all $\ell \in \{m+1,\ldots,k\}$.

A proof of Lemma C.1 is given in [20].

Proof. Based on m, defined in (53), we have two cases:

Case 1: $m = k \quad 1$.

Since $m \in J_j$, (52) implies

$$o_{T`}(b_j(\sigma_k)) \quad \bar{f}(T`) \quad \epsilon_{`;j}(\sigma_k), \tag{C.2}$$

for $\ell = 1,2,\ldots,k \quad 1$. Applying (50), we have

$$o_{T_m}(b_j(\sigma_k)) \quad \bar{f}(T_m) \quad \epsilon_{m;j}(\sigma_k) = \bar{f}(T_k) \tag{C.3}$$

From (54), we have $m+1 \in K_j$. Then the value of $\sigma\ (j)$ obtained from (48) and (49) (see Theorem 2) is $\sigma\ (j) = \sigma_k$. On the other hand, from (49) and (C.2), the value of $\sigma\ (j)$ according to (57) is also σ_k.

Case 2: $m < k \quad 1$.

Let us assume $\sigma\ (j)$ obtained from (57) is σ_n. We will show $\sigma\ (j)$ obtained from (49) and (48) will also be σ_n. In

this case, according to (54) and (57), if $\ell \; 2 \; f1, 2, \ldots, n \quad 1g$ then $\ell \; 2 \; J_j$ and $\ell < k \quad 1$. Applying Lemma C.1,

$$o_{T\cdot}(b_j(\sigma_n)) \quad \bar{f}(T\cdot) \quad \frac{W\ (b_j(\sigma_n); A_0) \quad T\cdot}{\rho b_j(\sigma_n)}(1 \quad \bar{f}(T\cdot)), \tag{C.4}$$

for $l = 1, 2, \ldots, n \quad 1$. On the other hand, using $n \; 2 \; K_j$, (54), and (50) we have

$$o_{T_n \;\; 1}(b_j(\sigma_n)) \quad \bar{f}(T_n \;\; 1) = \bar{f}(T_n) \quad \epsilon_n \;\; 1;j(\sigma_n) \tag{C.5}$$

According to (C.4) and (C.5), $\sigma\ (j)$ obtained from (49) and (48) will also be σ_n.

In both cases, we have shown that $\sigma\ (j)$ defined by (57) is the same as $\sigma\ (j)$ given in (48). Thus, the $(\sigma\ , \rho)$ regulators specified in Theorems 2 and 3 (respectively, Algorithms 2 and 3) produce the same output flow for a given input traffic flow. □

ACKNOWLEDGMENTS

The authors thank Yariv Ephraim for helpful discussions and the associate editor and anonymous reviewers for their constructive comments.

REFERENCES

[1] M. Kordi Boroujeny, B. L. Mark, and Y. Ephraim, "Stochastic traffic regulator for end-to-end network delay guarantees," in *Proc. IEEE Int. Conf. on Comm. (ICC)*, Jun. 2020.

[2] A. Kosta, N. Pappas, and V. Angelakis, "Age of Information: A New Concept, Metric, and Tool," *Foundations and Trends in Networking*, vol. 12, no. 3, pp. 162–259, 2017.

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

[4] ——, "A calculus for network delay. II. Network analysis," *IEEE Trans. Inf. Theory*, vol. 37, no. 1, pp. 132–141, Jan. 1991.

[5] B. L. Mark and G. Ramamurthy, "Real-time estimation and dynamic renegotiation of UPC parameters for arbitrary traffic sources in ATM networks," *IEEE/ACM Trans. Netw.*, vol. 6, pp. 811–827, Dec. 1998.

[6] C.-S. Chang, *Performance Guarantees in Communication Networks*. London: Springer-Verlag, 2000.

[7] M. Fidler and A. Rizk, "A guide to stochastic network calculus," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 1, pp. 59–86, 2015.

[8] Q. Yin, Y. Jiang, S. Jiang, and P. Y. Kong, "Analysis on generalized stochastically bounded bursty traffic for communication networks," in *Proc. IEEE Local Comput. Netw. (LCN)*, Nov. 2002, pp. 141–149.

[9] Y. Jiang, Q. Yin, Y. Liu, and S. Jiang, "Fundamental calculus on generalized stochastically bounded bursty traffic for communication networks," *Comput. Netw.*, vol. 53, no. 12, pp. 2011 – 2021, Aug. 2009.

[10] D. Starobinski and M. Sidi, "Stochastically bounded burstiness for communication networks," *IEEE Trans. Inf. Theory*, vol. 46, no. 1, pp. 206–212, Jan. 2000.

[11] O. Yaron and M. Sidi, "Performance and stability of communication networks via robust exponential bounds," *IEEE/ACM Trans. Netw.*, vol. 1, no. 3, pp. 372–385, Jun. 1993.

[12] C.-S. Chang, "Stability, queue length, and delay of deterministic and stochastic queueing networks," *IEEE Trans. Autom. Control*, vol. 39, no. 5, pp. 913–931, May 1994.

[13] Y. Jiang and Y. Liu, *Stochastic Network Calculus*. Springer-Verlag, 2008.

[14] O. Yaron and M. Sidi, "Generalized processor sharing networks with exponentially bounded burstiness arrivals," in *IEEE INFOCOM Proc.*, vol. 2, Jun. 1994, pp. 628–634.

[15] M. Kordi Boroujeny, B. L. Mark, and Y. Ephraim, "Tail-limited phase-type burstiness bounds for network traffic," in *53rd Conf. Info. Sciences and Systems (CISS)*, Baltimore, MD, Mar. 2019.

[16] H. Kobayashi and B. L. Mark, *System Modeling and Analysis: Foundations of System Performance Evaluation*. Pearson Education, Inc., 2009.

[17] M. Bladt and B. Nielsen, *Matrix-Exponential Distributions in Applied Probability*. New York, NY: Springer, 2017.

[18] M. Kordi Boroujeny, B. L. Mark, and Y. Ephraim, "Fitting network traffic to phase-type bounds," in *54rd Conf. Info. Sciences and Systems (CISS)*, Princeton, NJ, Mar. 2020.

[19] H. Kobayashi, B. L. Mark, and W. Turin, *Probability, Random Processes, and Statistical Analysis*. Cambridge University Press, 2012.

[20] M. Kordi Boroujeny and B. L. Mark, "Design of a stochastic traffic regulator for end-to-end network delay guarantees," arXiv:2008.07721 [cs.IT], Aug. 2020.

[21] L. Kleinrock and H. H. Goldstine, *Queueing Systems. Volume II: Computer Applications*. New York: John Wiley, 1976.

[22] W. Fischer and K. Meier-Hellstern, "The Markov-modulated Poisson process (MMPP) cookbook," *Perform. Eval.*, vol. 18, no. 2, pp. 149–171, 1993.

[23] G. Dán, V. Fodor, and G. Karlsson, "Packet size distribution: An aside?" in *Quality of Service in Multiservice IP Networks*, M. Ajmone Marsan, G. Bianchi, M. Listanti, and M. Meo, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 75–87.

[24] C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockell, T. Seely, and S. Diot, "Packet-level traffic measurements from the Sprint IP backbone," *IEEE Network*, vol. 17, no. 6, pp. 6–16, 2003.

Massieh Kordi Boroujeny (S'17) received the B.Sc degree in electrical engineering from the Shahid Bahonar University of Kerman in 2009, and the M.Sc. degree in electrical engineering from the Isfahan University of Technology in 2012. He is currently a Ph.D. candidate in the Dept. of Electrical and Computer Engineering at George Mason University and has been a Research Assistant since 2017. His research interests include network calculus, queueing theory, network resource allocation, and network measurement and analysis.

Brian L. Mark (S'91–M'95–SM'08) received the B.A.Sc. degree in computer engineering with a minor in mathematics from the University of Waterloo in 1991, and the Ph.D. degree in electrical engineering from Princeton University in 1995. From 1995 to 1999, he was a Research Staff Member with NEC USA, Princeton, New Jersey. In 1999, he was a Visiting Researcher with Télécom Paristech in Paris, France. In 2000, he joined George Mason University, where he is currently Professor of Electrical and Computer Engineering. He served as Acting Chair of the Department of Bioengineering from 2015–2017. His main research interests lie in the design and performance analysis of communication networks, wireless communications, statistical signal processing, and network security. Dr. Mark was an Associate Editor of the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY from 2006 to 2009.