Statistical Delay Guarantee of Virtual Clock *

Pawan Goyal
Networking and Distributed Systems Center
AT&T Labs - Research
Florham Park, NJ 07932

Harrick M. Vin
Department of Computer Sciences
University of Texas at Austin
Taylor Hall 2.124, Austin, TX 78712

Abstract

In this paper, we derive a statistical delay guarantee of the generalized Virtual Clock scheduling algorithm. We define the concept of an equivalent *fluid and packet source and prove a theorem that relates the departure time of a packet in a fluid FCFS multiplexor to its departure time in a packet multiplexor that uses generalized Virtual Clock algorithm for scheduling packets. This theorem enables us to use extant analyses of fluid FCFS multiplexors for providing statistical QoS guarantees in a network that employs the generalized Virtual Clock algorithm. We utilize the extant analysis of FCFS fluid multiplexors serving two-state on-off sources with exponentially distributed on and off durations to evaluate the increase in utilization yielded by our analysis technique. Our experiments demonstrate that for one of the source models employed in the literature, our technique can increase utilization by upto* 400% *compared to previously know statistical analysis methods.*

1 Introduction

Integrated services networks are required to support a variety of applications (e.g., audio and video conferencing, multimedia information retrieval, ftp, telnet, WWW, etc.) with a wide range of Quality of Service (QoS) requirements. Whereas continuous media applications such as audio and video conferencing require a network to provide QoS guarantees with respect to bandwidth, packet delay, and loss; applications such as telnet and WWW require low packet delay and loss. Throughput intensive applications like ftp, on the other

*A version of this paper will appear in the Proceedings of Real Time Systems Symposium, December 1998. This research was supported in part by IBM Faculty Development Award, Intel, the National Science Foundation (Research Initiation Award CCR-9409666), NASA, Mitsubishi Electric Research Laboratories (MERL), and Sun Microsystems Inc.

hand, require network resources to be allocated such that the throughput is maximized. A network meets these requirements primarily by appropriately *scheduling* its resources.

To appropriately schedule network bandwidth, several packet scheduling algorithms have been proposed in the literature [3, 4, 5, 7, 9, 11, 13, 14, 15, 21]. Furthermore, to enable the network to provide bounds on end-to-end delay as well as throughput guarantees to various flows, several analyses of these algorithms have been carried out [6, 8, 10, 13]. These techniques enable a network to provide deterministic bounds on QoS. Though these techniques are appropriate when the packet sources can be well characterized deterministically, they lead to underutilization of resources when the sources have significant statistical variations [12]. Analysis techniques for packet scheduling algorithms that enable a network to provide statistical guarantees and, thus, achieve higher utilization of resources have largely remained unexplored [1].

In this paper, we take a step towards addressing this problem by deriving a statistical delay guarantee of the generalized Virtual Clock scheduling algorithm [10]. We first model the sources as fluid processes and analyze the queuing behaviour of a FCFS fluid multiplexor. We then define the concept of an *equivalent* fluid and packet source and prove a theorem that relates the departure time of a packet in a fluid FCFS multiplexor to its departure time in a packet multiplexor that uses generalized Virtual Clock algorithm for scheduling packets. This theorem enables us to use extant analyses of fluid FCFS multiplexors for providing statistical QoS guarantees in a network that employs the generalized Virtual Clock algorithm. We utilize the analysis of FCFS fluid multiplexors serving two-state on-off sources with exponentially distributed on and off durations presented in [2] to evaluate the increase in utilization yielded by our analysis technique. Our experiments demonstrate that for one of the source models employed in the literature, our technique can increase utilization by upto 400% compared to previously know statistical analysis methods.

The rest of the paper is organized as follows. In Section 2, we derive the statistical delay guarantee of generalized Virtual Clock algorithm. We present the results of our experiments and related work in Sections 3 and 4, respectively. Finally, Section 5 summarizes the results of this paper.

2 Analysis of Generalized Virtual Clock

Several data and continuous media sources such as audio and video have an intrinsic rate of traffic generation that varies over time. Hence, they can be modeled as fluid processes generating fluid at time varying rates. Let the rate of fluid generation for source f at time t be $\widehat{R}_f(t)$. To determine delay and loss incurred by the fluid sources, consider a multiplexor serving Q sources. Let the scheduling algorithm at the multiplexor be FCFS and the stochastic behaviour of the Q sources be such that the queue length at the multiplexor at any time t, denoted by $\widehat{W}(t)$, is given as:

$$P\left(\widehat{W}(t) > \gamma\right) \leq G(\gamma)$$

Let us assume that function $G(\gamma)$ can be determined from the statistical characterization of the fluid sources. Hence, the delay and loss behaviour of the fluid sources is known. Since in computer networks sources transmit data as packets, our objective is to use function $G(\gamma)$ to determine QoS guarantees in an *equivalent* packet system. Clearly, this can be achieved only if appropriate packet scheduling algorithm is employed by the packet multiplexor. We choose generalized Virtual Clock (VC) as the packet scheduling algorithm. To achieve our objective, we:

- Define the concept of equivalent fluid and packet arrival processes (Section 2.1),
- Relate the departure time of a packet in the multiplexor employing the VC scheduling algorithm to the departure time of the last bit of the packet in the FCFS fluid multiplexor when the arrival processes at the packet and fluid multiplexors are equivalent (Section 2.2), and
- Derive the QoS guarantees for sources in a generalized VC multiplexor using the relationship between the fluid and packet system (Section 2.3).

In the following sections, we will use the following terminology. We will refer to the sequence of packets transmitted by a source to a destination as a flow. A flow is served by a sequence of packet switches (routers). We will refer to the output port of a switch as a server. The j^{th} packet of a flow and its arrival time will be denoted by p_f^j and $A(p_f^j)$, respectively.

2.1 Equivalent Fluid and Packet Processes

Consider a fluid source f generating fluid at rate $\widehat{R}_f(t)$. Then, a packet source can be considered equivalent to a fluid source if it generates a packet *exactly* at the time at which the first bit of the packet would be generated in the fluid source. This notion of equivalence is strong. Instead, we consider a packet source to be equivalent to the fluid source if it generates a packet *no later than* the time at which the first bit of the packet would be generated in the fluid source.

To formalize this notion of equivalence, we define a rate function for flow f, denoted by $R_f(t)$, based on *expected arrival time* of packets. The expected arrival time of a packet is the time at which the first bit of the packet would be generated in the fluid system. To formally define it, let r_f^j be the rate at which p_f^j is

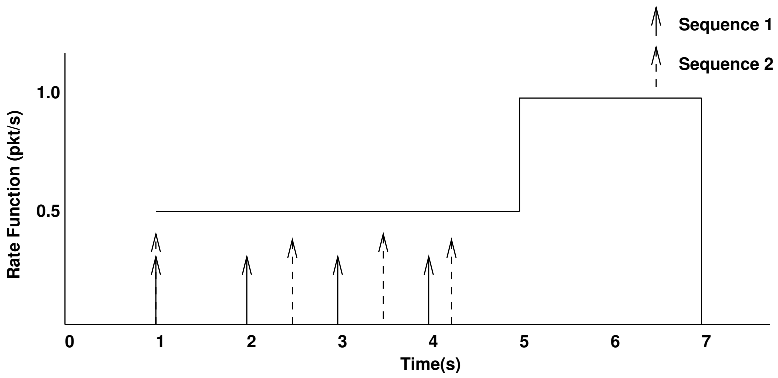

Figure 1: Equivalence of Fluid and Packet Sources

generated. Then, the expected arrival time of p_f^j, denoted by $EAT(p_f^j, r_f^j)$, is defined as:

$$EAT(p_f^j, r_f^j) = \max\left\{ A(p_f^j), EAT(p_f^{j-1}, r_f^{j-1}) + \frac{l_f^{j-1}}{r_f^{j-1}} \right\}$$

where $EAT(p_f^0, r_f^0) + \frac{l_f^0}{r_f^0} = 0$. We define $R_f(t)$ to be the rate at which p_f^j is generated in the time interval between its expected arrival time and the time at which its last bit would be generated in a fluid flow system. In a fluid flow system, if the first bit of p_f^j is generated at time $EAT(p_f^j, r_f^j)$, its last bit would be generated at time $EAT(p_f^j, r_f^j) + \frac{l_f^j}{r_f^j}$. Hence, $R_f(t)$ is formally defined as:

$$R_f(t) = \begin{cases} r_f^j & \text{if } \exists j \ni \left(EAT(p_f^j, r_f^j) < t \leq EAT(p_f^j, r_f^j) + \frac{l_f^j}{r_f^j} \right) \\ 0 & \text{otherwise} \end{cases}$$

We define a packet source and fluid source to be equivalent if $\widehat{R}_f(t) = R_f(t)$. Observe that $R_f(t)$ changes only at discrete time instants and does not vary during interval $[EAT(p_f^j, r_f^j), EAT(p_f^j, r_f^j) + \frac{l_f^j}{r_f^j}]$. Hence, $\widehat{R}_f(t) = R_f(t)$ only if fluid source changes rates at discrete time instants. We assume that this assumption is satisfied by the fluid source.

An important aspect of this notion of equivalence is that it only requires a packet of a flow to arrive no later than the time its first bit is generated in the fluid source; a packet may actually arrive much earlier than that. Thus, there are infinitely many packet arrival processes that would be considered equivalent to a fluid process. To illustrate this, consider an example fluid source that has rate $0.5pkt/s$ in $(1, 5]$, $1pkt/s$ in $(5, 7]$, and 0 elsewhere. Then, a packet source would be equivalent to the fluid source as long as packets 1,2,3, and 4 arrive by time 1,3,5, and 6, respectively. Figure 1 illustrates two packet arrival sequences that satisfy this assumption and thus are equivalent to the fluid source.

We now use this concept of equivalence to derive a new delay guarantee of generalized Virtual Clock algorithm.

2.2 Delay Guarantee of Generalized Virtual Clock

The generalized Virtual Clock (VC) scheduling algorithm, defined in [10], is a generalization of Virtual Clock presented in [22]. This generalization is based on the observation that since several continuous media and data sources generate data at time varying rates, a scheduling algorithm should be able to allocate variable rate to the packets of a flow. Specifically, it is defined as follows:

1. On arrival, packet p_f^j is stamped with virtual clock value, denoted by $VC(p_f^j, r_f^j)$, computed as:

$$VC(p_f^j, r_f^j) = EAT(p_f^j, r_f^j) + \frac{l_f^j}{r_f^j}$$

   where r_f^j is the rate assigned to p_f^j.

2. Packets are serviced in the increasing order of the virtual clock values.

Consider a VC server serving Q flows. It was shown in [10] that if C is the capacity of the server and $\sum_{n \in Q} R_n(t) \leq C$ for all t, then the VC server guarantees that the departure time of a packet, denoted by $L_{VC}(p_f^j)$, is given as:

$$L_{VC}(p_f^j) \leq VC(p_f^j, r_f^j) + \frac{l_{max}}{C} \quad (1)$$

where l_{max} is the maximum packet length served by a server. Our concept of *equivalence* between fluid and packet sources allows another interpretation of this guarantee. Let the rate assigned to a packet be the rate at which it is generated in the fluid system. Observe that when $\sum_{n \in Q} R_n(t) \leq C$ then buffer does not build up in a FCFS fluid server serving the *equivalent* Q fluid flows. Hence, the last bit of a packet departs at the time at which it arrives. Let $\widehat{L}(p_n^j)$ denote the time at which the last bit of packet p_f^j departs in the fluid FCFS server. Then, since the last bit of packet is generated at time $VC(p_f^j, r_f^j) = EAT(p_f^j, r_f^j) + \frac{l_f^j}{r_f^j}$, we conclude that:

$$L_{VC}(p_f^j) \leq \widehat{L}(p_f^j) + \frac{l_{max}}{C}$$

Hence, the departure time of a packet in a VC server is at most $\frac{l_{max}}{C}$ more than its departure time (i.e., the departure time of the last bit) in the FCFS fluid server serving equivalent fluid flows.

The above delay guarantee of a VC server and the relationship between the departure time of a packet in a VC server and equivalent FCFS fluid server holds only when $\sum_{n \in Q} R_n(t) \leq C$ for all t. When flows have significant statistical variations in their rate, i.e., $R_n(t)$ varies significantly over time, then ensuring

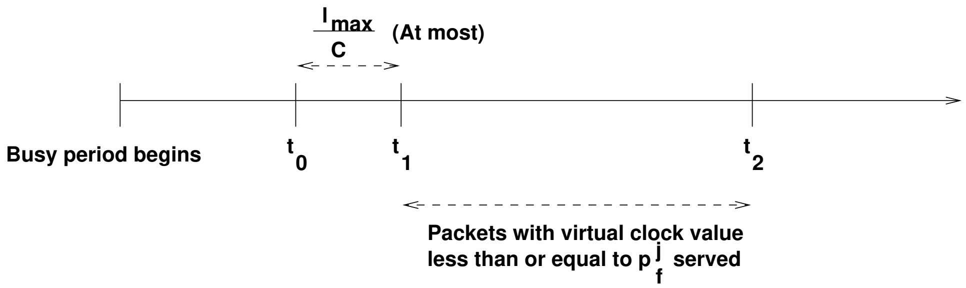

Figure 2: Timeline in the Virtual Clock server

that $\sum_{n \in Q} R_n(t) \leq C$ may lead to significant underutilization of resources. To achieve higher utilization of resources, it is desirable to derive the delay guarantee of a VC server when $\sum_{n \in Q} R_n(t) > C$. We achieve this objective in Theorem 1 by relating the departure time of packet in a VC server to its departure time in a FCFS fluid flow server serving equivalent fluid flows *without* putting any constraints on the rate functions.

In what follows, we will use the following terminology. We will consider the arrival and departure time of a packet in the fluid server to be the arrival and departure time of its first and last bit, respectively.

Theorem 1 *If the flows served by a VC server and a FCFS fluid server are equivalent, then the departure time of packet p_f^j in a VC server is given as:*

$$L_{VC}(p_f^j) \leq \widehat{L}(p_f^j) + \frac{l_{max}}{C}$$

Proof: Let $W(t_1, t_2)$ and $\widehat{W}(t_1, t_2)$ denote the work done by the VC server and FCFS fluid server, respectively, in $[t_1, t_2]$. Also, let $T(w)$ be the time taken to complete work w. Now, consider packet p_f^j. Define t_0, t_1, and t_2 as follows (see Figure 2):

- $t_2 = L_{VC}(p_f^j)$.
- t_0 and t_1: Let t_0 be the largest time instant less than t_2 in the busy period of the VC server in which p_f^j is served at which a packet with Virtual Clock value greater than that of p_f^j is scheduled. Let t_1 be the time at which such a packet finishes service. If such a packet does not exist, set t_0 and t_1 to the beginning of the busy period of the VC server.

Observe that packet p_f^j arrives only after t_0 in the VC server. This can be observed by considering two cases:

- t_0 is the beginning of the busy period: In this case it is trivially true.

- t_0 is not the beginning of the busy period: From the definition of t_0 we know that a packet with virtual clock value greater than that of p_f^j was scheduled at t_0. Since a VC server schedules packets in increasing order of virtual clock values, we conclude that p_f^j could not have arrived by t_0.

Hence, p_f^j arrives only after t_0 in the VC server. Since the fluid and packet arrival processes are equivalent and a packet can arrive only earlier in the packet system, we conclude that p_f^j can arrive only after t_0 in the fluid system. Hence, $t_0 < \widehat{L}(p_f^j)$. Therefore, we get:

$$t_0 + T(\widehat{W}(t_0^+, \widehat{L}(p_f^j))) \le \widehat{L}(p_f^j)$$

Since $T(w) \ge \frac{w}{C}$, we get:

$$t_0 + \frac{\widehat{W}(t_0^+, \widehat{L}(p_f^j)))}{C} \le \widehat{L}(p_f^j) \tag{2}$$

To determine the relationship between $\widehat{L}(p_f^j)$ and $L_{VC}(p_f^j)$, our objective now is to relate $\widehat{W}(t_0^+, \widehat{L}(p_f^j))$ to variables in the VC server. To do so, we relate $\widehat{W}(t_0^+, \widehat{L}(p_f^j)))$ to $W(t_1, t_2)$. Observe that:

- *Packets served by the VC server in* $[t_1, t_2]$ *are served by the fluid server after* t_0: First, observe that packets served in $[t_1, t_2]$ arrive only after t_0 in the VC server. Consider two cases:

  - t_0 is the beginning of the busy period: The claim holds trivially in this case.
  - t_0 is not the beginning of the busy period: In this case, from the definition of t_0 we know that a packet with virtual clock value greater than the virtual clock values of the packets served in $[t_1, t_2]$ was scheduled at t_0. Since a VC server serves packets in increasing order of virtual clock values, the packets served in $[t_1, t_2]$ could not have arrived by t_0.

  Hence, packets served in $[t_1, t_2]$ in the VC server arrive only after t_0 in the VC server. Since fluid and packet processes are equivalent, a packet can arrive only earlier in a VC server than in the fluid system. Hence, we conclude that the packets served by the VC server in $[t_1, t_2]$ arrive only after t_0 in the fluid server also and are consequently served after t_0.

- *Packets served by the VC server in* $[t_1, t_2]$ *are served by the fluid server by* $\widehat{L}(p_f^j)$: In the fluid arrival process, the last bit of a packet is generated at time equal to the Virtual Clock value of the packet. Since the fluid multiplexor is FCFS, we conclude that the packets depart the fluid server in the order of their Virtual Clock values (packets with equal virtual clock values depart simultaneously). Hence, all packets with Virtual Clock value less than or equal to that of p_f^j will be served in the fluid server by $\widehat{L}(p_f^j)$. Since all the packets served in time interval $[t_1, t_2]$ by VC server have virtual clock value less than or equal to that of p_f^j, we conclude that they will be served by the fluid server by $\widehat{L}(p_f^j)$.

Thus, we conclude that the packets served by the VC server in $[t_1, t_2]$ will be served by the fluid server in $[t_0^+, \widehat{L}(p_f^j)]$. Therefore, we get:

$$W(t_1, t_2) \leq \widehat{W}(t_0^+, \widehat{L}(p_f^j)) \tag{3}$$

Using (3) to substitute in (2), we get:

$$t_0 + \frac{W(t_1, t_2)}{C} \leq \widehat{L}(p_f^j)$$

Since the VC server is busy in $[t_1, t_2]$, $W(t_1, t_2) = (t_2 - t_1)C$. Hence,

$$\begin{aligned} t_0 + \frac{(t_2 - t_1)C}{C} &\leq \widehat{L}(p_f^j) \\ t_2 &\leq \widehat{L}(p_f^j) + (t_1 - t_0) \end{aligned}$$

Since $t_2 = L_{VC}(p_f^j)$ and $t_1 - t_0 \leq \frac{l_{max}}{C}$, we get:

$$L_{VC}(p_f^j) \leq \widehat{L}(p_f^j) + \frac{l_{max}}{C}$$

■

Note that we have made no assumptions about the rate functions in Theorem 1. The Theorem yields (1) when $\sum_{n \in Q} R_n(t) \leq C$ but continues to hold even when $\sum_{n \in Q} R_n(t) > C$. Thus, it subsumes the previous theorems on Virtual Clock. Furthermore, to the best of our knowledge, this is the only result that relates the departure time of a packet in a fluid FCFS server to that in a packet server.

The relationship between the fluid analysis and the packet analysis only requires $R_n(t) = \widehat{R}_n(t)$. As we had observed earlier, there are infinitely many packet arrival process that satisfy this assumption. Thus, the relationship between fluid and packet analysis for a Virtual Clock server holds for infinitely many realizations of a fluid process. This weaker notion of equivalence is desirable in packet networks because due to variation in packet processing times as well as granularity of timers at the host operating system, packets may not be generated *exactly* at the time they are required to be. Furthermore, this weaker notion of equivalence provides protection from flows that misbehave either intentionally or due to faulty hardware.

The above theorem can be misinterpreted as being the same as Theorem 1 of [13] that relates the departure time of a packet in a Packet-by-packet Generalized Processor Sharing (PGPS) server to its departure time in a fluid Generalized Processor Sharing (GPS) server. However, there are several important differences. To elucidate the differences, let us briefly review Theorem 1 of [13]. The theorem states that if $L_{PGPS}(p_f^j)$ is the departure time of packet p_f^j in a PGPS server and $L_{GPS}(p_f^j)$ is its departure time in a GPS server under the *same arrival sequence* of packets, then:

$$L_{PGPS}(p_f^j) \leq L_{GPS}(p_f^j) + \frac{l_{max}}{C} \tag{4}$$

There are two important aspects of (4).

- (4) is derived under the assumption that, in a GPS server, fluid amount equivalent to the length of a packet arrives exactly at the time instant at which the packet arrives in the PGPS server. This is in contrast to our notion of equivalence between fluid and packet sources in which the packet can arrive in the packet system significantly before it starts arriving in the fluid system.

- The fluid server uses the GPS scheduling algorithm. This is in contrast to Theorem 1 where the fluid server is a FCFS server.

We demonstrate the importance of these differences by illustrating through an example that a result which is analogous to (4) does not hold for Virtual Clock. Let $L_{F-VC}(p_f^j)$ be the departure time of a packet in a fluid VC server when fluid equivalent to the length of a packet arrives exactly at the time at which the packet arrives. Then, a claim analogous to (4) would be:

$$L_{VC}(p_f^j) \leq L_{F-VC}(p_f^j) + \frac{l_{max}}{C} \tag{5}$$

The following example demonstrates that this claim is incorrect.

Example 1 *Consider a server with capacity $3bits/s$. Let it serve three flows each with rate of $1bit/s$. Let the length of packets transmitted by flows 1 and 3 be 1 bit and that by flow 2 be 2 bits. Let flows 1 and 2 send 9 and 7 packets, respectively, at time 0. Let flow 1 send an additional packet at time 7^+, and flow 3 infinite number of packets at time $7\frac{2}{3}$. Now, consider the departure time of p_1^{10} in fluid Virtual Clock and Virtual Clock servers.*

- *Fluid Virtual Clock: The virtual clock values of p_2^7 and p_1^{10} are 14 and 10, respectively. Hence, the fluid Virtual Clock server will service p_1^{10} before p_2^7. Consequently,*

$$L_{F-VC}(p_1^{10}) = 7^+ + \frac{1}{3} \tag{6}$$

- *Virtual Clock: p_1^{10} can not be scheduled until p_2^7 finishes at time $7\frac{2}{3}$. But at time $7\frac{2}{3}$, packets from flow 3 arrive. The virtual clock values for flow 3 packets are given as:*

$$VC(p_3^k, 1) = 7\frac{2}{3} + k \tag{7}$$

Hence, we conclude that virtual clock value of packet p_3^2 is less than virtual clock value of p_1^{10}. Consequently, packets p_3^1 and p_3^2 will be served before p_1^{10} is served. Hence, we get:

$$L_{VC}(p_1^{10}) = 7\frac{2}{3} + \frac{2}{3} \tag{8}$$

From (6) and (8) we conclude that (5) does not hold.

The example can be extended to demonstrate that the departure time in the packet system can be arbitrarily away from the departure time in the fluid system. This example also demonstrates that our concept of equivalence between fluid and packet sources is central to the derivation of Theorem 1.

2.3 Determining QoS Guarantees for Flows

Our objective is to determine the QoS guarantees for flows being served by a generalized VC server. As outlined before, we achieve this objective by:

- Using fluid flow analysis of FCFS fluid flow server serving equivalent flows to determine the QoS guarantees for the flows, and
- Utilizing Theorem 1 to determine the QoS guarantees for the generalized VC server.

Consider a FCFS fluid flow server serving flows that are equivalent to the flows in a generalized VC server. Let infinite buffer be available at the fluid server. Furthermore, let the behaviour of flows be such that $P\left(\widehat{W}(t) > \gamma\right) \leq G(\gamma)$. Let us assume that the function $G(\gamma)$ is known. Now, consider an equivalent VC server. Since the VC server will have finite buffer in practice, let it serve only packets for which

$$L_{VC}(p_f^j) \leq VC(p_f^j, r_f^j) + d + \frac{l_{max}}{C} \tag{9}$$

where d depends on the available buffer space, denoted by B, at the server (one possible relationship between d and B is $B = C \cdot d$). Thus, in the VC server packets for which (9) is not satisfied are dropped. Hence, the loss rate for a flow is given by the probability of (9) not being satisfied. Our objective now is to employ the fluid analysis and Theorem 1 to determine the loss rate.

Observe that in the fluid server the last bit of a packet is generated at its virtual clock value. Hence, in the fluid server:

$$\widehat{L}(p_f^j) \leq VC(p_f^j, r_f^j) + \frac{\widehat{W}(t_0)}{C} \tag{10}$$

where $t_0 = VC(p_f^j, r_f^j)$. From Theorem 1 we know that

$$L_{VC}(p_f^j) \leq \widehat{L}(p_f^j) + \frac{l_{max}}{C} \tag{11}$$

Using (10) and (11), we get:

$$L_{VC}(p_f^j) \leq VC(p_f^j, r_f^j) + \frac{\widehat{W}(t_0)}{C} + \frac{l_{max}}{C} \tag{12}$$

From (12) and (9), we conclude that if we drop packet p_f^j when it does not satisfy (9), then $\frac{\widehat{W}(t_0)}{C} > d$. Thus, if we assume that all the flows are independent and the state of the queue observed by a flow is the same as the time average of the queue, then the probability of loss of a packet, denoted by q, can be approximated as :

$$q = P\left(\frac{\widehat{W}(t)}{C} > d\right)$$

Since $P\left(\widehat{W}(t) > \gamma\right) \leq G(\gamma)$, we get:

$$q = G(C \cdot d) \tag{13}$$

Thus, given d and $G(\gamma)$, the loss rate for a flow can be determined using (13). Furthermore, a bound on the departure time of packets is determined using (9).

Our objective now is to determine the maximum delay incurred by packets at the VC server using (9). Recall that Theorem 1 and consequently (9) holds for infinitely many arrival processes. To determine the maximum packet delay incurred by packets in the VC server, we need to constrain the packet arrival process. Hence, we assume that the packets arrive at their expected arrival time. Thus,

$$VC(p_f^j, r_f^j) - A(p_f^j) = VC(p_f^j, r_f^j) - EAT(p_f^j, r_f^j) = \frac{l_f^j}{r_f^j} \tag{14}$$

Hence, using (9) and (14), we conclude that delay for packet p_f^j, denoted by d_f^j, is given as:

$$d_f^j \leq \frac{l_f^j}{r_f^j} + d + \frac{l_{max}}{C}$$

Hence, the maximum delay incurred by packets of flow f is $d_f^{max} = \max\{d_f^j\}$ where the maximum is over all the packets of the flow. Thus, using the analysis of a fluid FCFS server, we can determine the delay and loss guarantees for a flow in a generalized VC server. Our objective now is to extend this analysis to a network of servers.

We extend the above single server analysis to multiple servers by employing the methodology in [20]. We assume that all the servers in the network employ generalized Virtual Clock as the packet scheduling algorithm. Furthermore, we assume that the mechanisms required for allocating variable rate in a VC server to the packets of a flow (such as the fast reservation protocol in [16]) are available. Finally, we assume that jitter controller as in [20] is employed at each server to reconstruct the traffic pattern of each flow at each server on the path. Let there be K servers on the path of flow f and i^{th} server on the path be server i. Also, let q^i and $d_f^{max,i}$ be the packet loss probability and the maximum packet delay at server i determined using the single server analysis. Then, as shown in [20], the maximum end-to-end delay is $\sum_{i=1}^{i=K} d_f^{max,i}$ and the packet loss probability is $\sum_{i=1}^{i=K} q^i$.

In the next section, we evaluate the increase in utilization yielded by our analysis.

3 Experimental Evaluation

We determine the increase in utilization when a network provides statistical guarantee to on/off sources with exponentially distributed on and off periods. An on/off source is a two state process that is either in an "on" state or in an "off" state. It generates data at a constant rate R in the on state and no data in the off state. The time spent by it in on and off state are exponentially distributed. Thus, if I_{on} and I_{off} denote the average time spent by the source in on and off state, respectively, then the source is completely characterized by the tuple (I_{on}, I_{off}, R). We choose two specific values for the tuple and term the resultant sources EXP1 and EXP2:

- EXP1: For this source, $I_{on} = 312ms$, $I_{off} = 325ms$, and $R = 64Kb/s$. This model has been considered appropriate for audio source [12].
- EXP2: For this source, $I_{on} = 9.76ms$, $I_{off} = 90ms$, and $R = 1Mb/s$. This is one of the source models used in [12] and may be appropriate for some data sources.

In both the models, we assume that only complete packets are generated, i.e., if the source would make a transition to off state before a complete packet can be generated, no packet is generated.

For ease of analysis, we consider networks that only serve either EXP1 or EXP2 sources but not both. Furthermore, we assume that the load on each server on the path of a flow is same, i.e, each server serves same number of flows. In such a case, if d_f^{max} and q are the maximum delay and packet loss probability at a single server, then the maximum end-to-end delay is Kd_f^{max} and the packet loss probability is Kq. Thus, we can determine the end-to-end delay and loss probability from the single server analysis.

To determine the loss probability and maximum delay for a flow in a VC server, we require the function $G(\gamma)$ for a FCFS fluid server serving equivalent fluid flows. The steady state distribution of queue length of a FCFS fluid server when all the sources are identical on/off fluid sources with exponentially distributed periods has been derived in [2]. Thus, function $G(\gamma)$ such that $P(\widehat{W}(t) > \gamma) \leq G(\gamma)$ when $t \rightarrow \infty$ is known. As argued in [18], $P(\widehat{W}(t) > \gamma) \leq G(\gamma)$ even when t is finite. Hence, we can determine the delay and loss probability in the fluid FCFS server. Thus, using the analysis presented in Section 2.3 we can use it to determine the delay and loss probability in the VC server.

We now present our evaluation for EXP1 and EXP2 sources when the capacity of each of the servers in the network is $10Mb/s$ and the packet size is 512 bits.

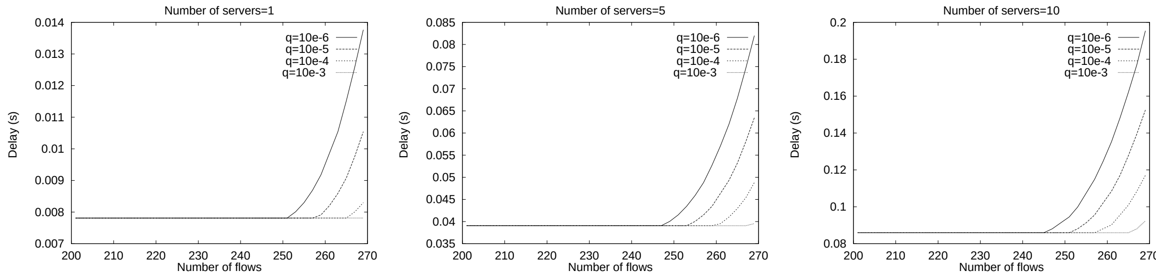

Figure 3: EXP1 source: (a) 1 Server, (b) 5 Servers, and (c) 10 Servers

- EXP1: Figures 3(a), 3(b), and 3(c) plot the end-to-end delay versus number of flows for 1, 5, and 10 server paths, respectively, for EXP1 flows. As Figure 3(a) demonstrates, the maximum delay for 269 flows is 13.8 ms when the desired loss probability is 10^{-6} and there is one server on the path. This illustrates that our analysis matches the performance of the measurement based admission control algorithm in [12] which accepts 250 flows on an average. The figures also show that the end-to-end delay increases with increase in the number of servers. For example, the delay increases to 81.9 ms for 269 flows when the number of servers increases to 5.

  To validate the analytical predictions, we simulated a VC server serving 269 EXP1 flows for 1000 seconds. In conformance with the predictions, no packet was lost in the simulation.

- EXP2: Figures 4(a), 4(b), and 4(c) plot the end-to-end delay versus number of flows for 1, 5, and 10 server paths, respectively, for EXP2 flows. As Figure 4(a) demonstrates, the maximum delay for 79 flows is 41 ms when the desired loss probability is 10^{-6} and there is one server on the path. This illustrates that our analysis matches the performance of the measurement based admission control algorithm in [12] which accepts 75 flows on an average. Also, the maximum delay incurred by a packet was found to be 42ms through simulations in [12]. Thus, the analytical predictions conform with simulation observations reported in the literature. The figures also show that as in the case of EXP1, the end-to-end delay increases with increase in the number of servers. For example, the delay increases to 221.7 ms for 79 flows when the number of servers increases to 5.

  To validate the analytical predictions, we conducted several simulations. In each simulation, a VC server served fixed number of EXP2 flows. Each simulation was conducted for 1000 seconds and packets that violated (9) were dropped. For a given number of flows, the delay term d in (9) was determined from Figure 4(a) for loss probability of 10^{-3}. Figure 5 plots the experimentally observed loss rates averaged over all the flows for varying number of flows. Observe that the average loss rate is significantly below 10^{-3}. Furthermore, we found that the loss rate for all the flows was below 10^{-3}

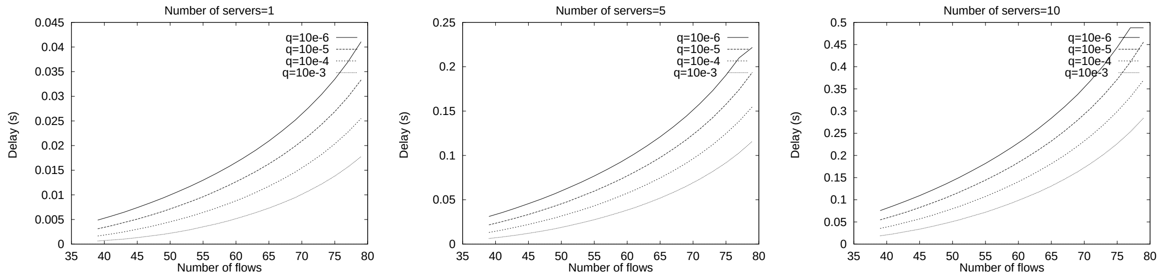

Figure 4: EXP2 source: (a) 1 Server, (b) 5 Servers, and (c) 10 Servers

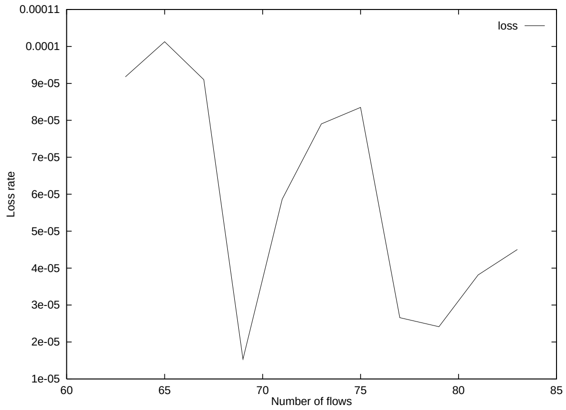

Figure 5: The loss rate for EXP2 sources determined through simulations

(in fact the loss rates for all the flows was significantly below 10^{-3}). Hence, we conclude that the simulations confirm the analytical predictions.

If the network were to provide deterministic guarantees, then at most 160 EXP1 flows ($\frac{10Mbs}{64Kb/s} = 160$) and 10 EXP2 flows ($\frac{10Mbs}{1Mb/s} = 10$) could have been accepted regardless of the maximum acceptable end-to-end delay and the number of servers on the path. In contrast, from Figures 3(b) and 4(b) we conclude that our analysis allows the network to accept 269 EXP1 flows and 79 flows when the number of servers on the path is 5, the desired loss probability is 10^{-6}, and maximum end-to-end delay of 81.9 ms and 221.7 ms is acceptable for EXP1 and EXP2 flows, respectively. This demonstrates that our analysis enables a network to achieve significantly higher utilization when statistical guarantees are acceptable.

4 Related Work

Statistical guarantees of Generalized Processor Sharing (GPS) when the sources have *exponentially bounded burstiness* have been derived in [18, 19, 23]. Our approach is different from the approach taken in [19, 23]. Whereas [19, 23] focus on analyzing the behaviour of a fluid GPS server, we leverage off existing analysis for FCFS fluid servers and focus on determining the performance guarantees of an equivalent packet system. In contrast to a VC server, extant analysis of fluid FCFS servers can not be employed for PGPS server. Other differences between the guarantees of a PGPS and a VC server have been discussed at length in Section 2.2. A direct comparison with the achievable utilization yielded using analysis technique presented in [23] does not seem to be possible.

An approach for providing statistical QoS guarantees to flows in a VC server has also been presented in [17]. The approach in [17] employs the *deterministic* delay guarantee of a VC server to provide statistical guarantees as follows. It assumes that packets that cause $\sum_{n \in Q} R_n(t)$ to exceed C are dropped. Hence, the packet loss probability, denoted by q, is approximated as:

$$q = P\left(\sum_{n \in Q} R_n(t) > C\right) \tag{15}$$

Since in a fluid server with zero buffer loss occurs whenever $\sum_{n \in Q} R_n(t)$ exceeds C, we term this the *Zero Buffer* approach. The loss probability q is determined using the probability distribution of the rate function of each flow. The single server analysis is extended to multiple servers in a manner analogous to ours. The key difference between this and our approach is that whereas in this approach packets that cause $\sum_{n \in Q} R_n(t)$ to exceed C are dropped, we buffer packets that may lead to violation of this condition and drop only packets that violate (9). Thus, our approach utilizes additional buffer in servers and, hence, is termed *Finite Buffer* approach. The advantage of Finite Buffer approach over Zero Buffer is that it provides additional flexibility to a network and enables it to achieve higher utilization when the maximum acceptable delay for flows is higher than the minimum that can be guaranteed. To demonstrate this, we determine the number of EXP1 and EXP2 flows that can be admitted using the Zero Buffer approach.

The probability distribution of the rate function for EXP1 and EXP2 sources is determined as follows. The probability of being in on state, denoted by p_{on} is $p_{on} = \frac{I_{on}}{I_{on}+I_{off}}$, and being in off state, denoted by p_{off}, is $1 - p_{on}$. Thus, $P(R_f(t) = R) = p_{on}$ and $P(R_f(t) = 0) = p_{off}$. Hence, the distribution of the rate function of the EXP1 and EXP2 sources is known. Thus, the maximum number of EXP1 and EXP2 flows that can be admitted for a given packet loss probability in Zero Buffer approach can be determined using (15) and its extension to the multiple server case. Figures 6(a) and 6(b) plot the number of flows admitted for different number of servers on the path and different end-to-end loss probability for EXP1 and EXP2 sources, respectively. The figures demonstrate that 251 EXP1 and 20 EXP2 flows are admitted when the desired loss probability is 10^{-6} and there is one server on the path. The number of admissible EXP1 and EXP2 flows decreases to 246 and 18, respectively, when the number of servers increases to 5.

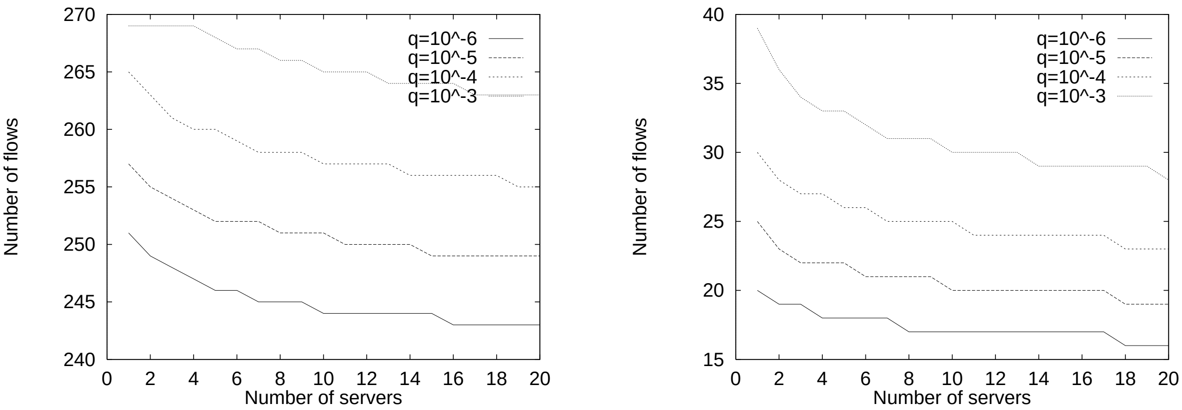

Figure 6: (a) EXP1 Source (b) EXP2 source

Figures 3, 4, and 6 show that the achievable utilization is always higher using the Finite Buffer approach for both EXP1 and EXP2 sources. To observe the increase in utilization, let the number of servers on the path and the desired packet loss probability be 5 and 10^{-6}, respectively. Then the number of admissible EXP1 flows when the delay requirement is 82 ms is 269 and 246 in Finite Buffer and Zero Buffer, respectively. Similarly, the number of admissible EXP2 flows when the delay requirement is 222 ms is 79 and 18 in Finite Buffer and Zero Buffer, respectively. The increase in utilization is higher in EXP2 than in EXP1 as EXP2 source has higher peak to average ratio and, hence, is more bursty. Thus, the additional flexibility provided by our approach (Finite Buffer) enables a network to achieve significantly higher utilization than the approach (Zero Buffer) in [17].

5 Conclusions

In this paper, we derived a statistical delay guarantee of the generalized Virtual Clock scheduling algorithm. We defined the concept of an *equivalent* fluid and packet source and proved a theorem that relates the departure time of a packet in a fluid FCFS multiplexor to its departure time in a packet multiplexor that uses generalized Virtual Clock algorithm for scheduling packets. This theorem enables us to use extant analyses of fluid FCFS multiplexors for providing statistical QoS guarantees in a network that employs the generalized Virtual Clock algorithm. We utilized the analysis of FCFS fluid multiplexors serving two state on-off sources with exponentially distributed on and off durations presented in [2] to evaluate the increase in utilization yielded by our analysis technique. Our experiments demonstrate that for one of the source models employed in the literature, our technique can increase utilization by upto 400% compared to previously know statistical analysis methods.

References

[1] Workshop on Packet Scheduling held in conjunction with SIGCOMM'96. August 1996.

[2] D. Anick, D. Mitra, and M.M. Sondhi. Stochastic Theory of a Data Handling System with Multiple Sources. *Bell Systems Technical Journal*, 61:1871–1894, 1982.

[3] J.C.R. Bennett and H. Zhang. WF^2Q: Worst-case Fair Weighted Fair Queuing. In *Proceedings of INFOCOM'96*, pages 120–127, March 1996.

[4] J.C.R. Bennett and H. Zhang. Hierarchical Packet Fair Queuing Algorithms. *IEEE/ACM Transactions on Networking*, 5(5), October 1997.

[5] N. Figuera and J. Pasquale. Leave-in-Time: A New Service Discipline for Real-Time Communication in a Packet-Switching Data Network. In *Proceedings of ACM SIGCOMM'95*, pages 207–218, 1995.

[6] L. Georgiadis, R. Guerin, V. Peris, and K.N. Sivarajan. Efficient Network QoS Provisioning Based on per Node Traffic Shaping. In *Proceedings of INFOCOM'96*, pages 102–110, March 1996.

[7] P. Goyal. *Packet Scheduling Algorithms for Integrated Services Networks*. PhD thesis, Department of Computer Sciences, The University of Texas at Austin, 1997.

[8] P. Goyal, S. S. Lam, and H. M. Vin. Determining End-to-End Delay Bounds In Heterogeneous Networks. *ACM/Springer-Verlag Multimedia Systems Journal*, 5(3):157–163, May 1997. Also appeared in the Proceedings of the Workshop on Network and Operating System Support for Digital Audio and Video, Pages 287-298, April 1995.

[9] P. Goyal and H. M. Vin. Fair Airport Scheduling Algorithms. In *Proceedings of the Workshop on Network and Operating System Support for Digital Audio and Video*, pages 273–282, May 1997.

[10] P. Goyal and H. M. Vin. Generalized Guaranteed Rate Scheduling Algorithms: A Framework. *IEEE/ACM Transactions on Networking*, 5(4):161–171, August 1997.

[11] P. Goyal, H. M. Vin, and H. Cheng. Start-time Fair Queuing: A Scheduling Algorithm for Integrated Services Packet Switching Networks. *IEEE/ACM Transactions on Networking*, 5(5), October 1997.

[12] S. Jamin, P. Danzig, S. Shenker, and L. Zhang. A Measurement Based Admission Control Algorithm for Integrated Services Packet Networks. In *Proceedings of ACM SIGCOMM'95*, pages 2–13, 1995.

[13] A.K. Parekh. *A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks*. PhD thesis, Department of Electrical Engineering and Computer Science, MIT, 1992.

[14] D. Stiliadis. *Traffic Scheduling in Packet-Switched Networks: Analysis, Design and Implementation*. PhD thesis, Department of Computer Science and Engineering, University of California at Santa Cruz, 1996.

[15] S. Suri, G. Varghese, and G. Chandramenon. Leap Forward Virtual Clock: A New Fair Queuing Scheme with Guaranteed Delays and Throughput fairness. In *Proceedings of INFOCOM'97*, April 1997.

[16] J. S. Turner. Managing Bandwidth in ATM Networks with Bursty Traffic. *IEEE Network*, 6(5):5–58, September 1992.

[17] G. Xie and S. S. Lam. Real-Time Block Transfer Under a Link Sharing Hierarchy. In *Proceedings of INFOCOM'97*, April 1997.

[18] O. Yaron and M. Sidi. Performance and Stability of Communication Networks via Robust Exponential Bounds. In *IEEE/ACM Transactions on Networking*, volume 1, pages 372–385, 1993.

[19] O. Yaron and M. Sidi. Generalized Processor Sharing Networks with Exponentially Bounded Burstiness Arrivals. In *Proceedings of INFOCOM'94*, 1994.

[20] H. Zhang. *Service Disciplines for Integrated Services Networks*. PhD thesis, Department of Electrical Engineering and Computer Science, University of California at Berkeley, 1993.

[21] H. Zhang. Service Disciplines For Guaranteed Performance Service in Packet-Switching Networks. *Proceedings of the IEEE*, 83(10), October 1995.

[22] L. Zhang. VirtualClock: A New Traffic Control Algorithm for Packet Switching Networks. In *Proceedings of ACM SIGCOMM'90*, pages 19–29, August 1990.

[23] Z. L. Zhang, D. Towsley, and J. Kurose. Statistical Analysis of Generalized Processor Sharing Discipline. In *Proceedings of ACM SIGCOMM'94*, pages 68–77, 1994.