Bandwidth Estimation for Video Streaming Under Percentile Delay, Jitter, and Packet Loss Rate Constraints Using Traces

Bushra Anjum (Corresponding Author)
banjum@ncsu.edu

Computer Science Department
NC State University
Raleigh, USA
Phone: (919) 515-2041
Fax: (919) 515-5039

and

Harry Perros
hp@csc.ncsu.edu

Computer Science Department
NC State University
Raleigh, USA

Abstract:

We present and use a CPU-efficient activity-based simulation model to calculate the sojourn time of a packet and the packet loss rate in a tandem queueing network that depicts the path of a video flow. The video flow is characterized by a packet trace. Background traffic, also characterized by a trace, is allowed in the tandem queueing network. In our analysis we used real video traces (Telepresence, WebEx, Jabber) and also generalized our results using traces generated by a theoretical model of a video arrival process depicted by a Markovian Arrival Process. Using this simulation model we calculate the bandwidth required for a video flow, so that a given set of constraints for the percentile end-to-end delay, jitter, and packet loss rate are satisfied. We also show that the bandwidth required for n identical video streams that follow the same path through an IP network, so that that the end-to-end percentile delay remains the same, is a linear function of n. Further, it is experimentally depicted that for infinite-capacity queues the bandwidth required to satisfy the percentile end-to-end delay constraint also satisfies the jitter constraint. And for finite-capacity queues, the bandwidth required to satisfy both the percentile end-to-end delay and the packet loss rate constraints also satisfies the pair of jitter and packet loss rate constraints.

Key words

Activity-based simulation, bandwidth estimation, video traces, percentile end-to-end delay, packet loss rate

1. Introduction

Video traffic is widely expected to account for a large portion of the traffic in future wired and wireless networks. Best practices dictate that in order to guarantee a good QoS for interactive video, the end-to-end delay has to be less than 150 msec (though delays up to 200 msec can also be tolerated), the jitter has to be less than 30 msec, and the packet loss rate less than 1%, see [1]. Interestingly enough, little is known about how much bandwidth should be allocated to an interactive video so that the above three QoS metrics can be satisfied. In this paper, we address this issue using video traces. That is, for a given video trace we calculate the required bandwidth that has to be allocated on each link along the path of the video flow, so that the above three QoS metrics are satisfied. We note that the work presented here is applicable to any trace and any combination of values of the above three QoS metrics.

We formulated this problem as a tandem queueing network consisting of N queues where each queue represents the output port of a router along the path of the video flow. More specifically, it represents the particular DiffServ queue that serves all video flows, such as queue AF41. The video flow, expressed by a trace of IP packets, is offered to the first queue, the output of which is offered to the second queue, and so on. We refer to this traffic as the *tagged stream*. In addition, we assume a background arrival process to each queue, referred to as the *background stream*, which represents other video flows also served by the same DiffServ queue. The service time at each queue is constant, proportional to the packet length of each packet in the trace. For this queueing network, we obtain the end-to-end delay, jitter, and packet loss rate, given that both tagged and background streams are described by traces.

We note that the analysis of a tandem queueing network with tagged and background arrival processes is difficult, and to the best of our knowledge there are no exact solutions. One aspect of this problem that has been studied extensively is the inter-departure time of the tagged stream from a queue that also serves background traffic. Dasu [7] obtained a closed–form expression of the Laplace transform of the inter–departure time of the tagged traffic in a two–class single server queueing system where the tagged arrival process is a generalized phase process, the background arrival process is Poisson, and the service time follows a phase–type distribution. Several approximations have also been reported under a variety of assumptions. In Kumaran et al. [13], the tagged and the background arrival processes were assumed to be matrix exponential (ME), and the service time distribution was also an ME. The authors obtained an approximation for the tagged departure process. More recently, Geleji and Perros [9] obtained an exact numerical solution of the inter-departure time of a tagged process from a single queue, assuming that both the tagged arrival process and the background arrival process are MMPP, and that the service time is exponentially distributed. Under more general assumptions, Geleji and Perros [10] also gave an analytic upper bound of the tagged inter-departure time from a tandem queueing network of any number of queues.

In [11], Ioannis and Stavrakakis proposed a queueing system to study the distortion induced in a tagged ATM stream. A discrete-time analysis in the M/G/1 paradigm yielded numerical results for cell delay, delay jitter, and inter-departure time probability distributions of an ATM multiplexer. Similarly, Conti et al. [5] evaluated the impact of temporal and spatial correlations on the end-to-end performance of a tagged traffic stream which can be due to background traffic or partial commonality in the routing path. They proposed a binary queueing activity indicator to provide for a simple mechanism to capture these correlations. Feng and Chang [8] used single-queue decomposition to analyze a tandem queueing network with general service times where the arrival process to the first queue consisted of multiple heterogeneous MMPPs. They proposed two approximation schemes for calculating the mean end-to-end delay for a single (tagged) MMPP stream. Central to the proposed schemes is the calculation of the departure process of the tagged MMPP stream by a two-state MMPP calculated using the first three moments of the inter-departure time and the lag 1 autocorrelation of the successive inter-departure times. Anjum et al [2] considered a tandem queueing network of infinite capacity queues with a two-state MMPP tagged arrival process and exponentially distributed service times. No background traffic was considered. They showed experimentally that the approximation of the departure process by a two-state MMPP proposed in Feng and Chang [8] does not give good results due to the fact that it does not capture well the autocorrelation structure of the departure process. Instead of nodal decomposition, the authors proposed an efficient and accurate approximation method for calculating a given percentile of the end-to-end delay using an exact upper and lower bound. Using this method, they calculated the required bandwidth that should be allocated on each link along the path to satisfy a given percentile of the end-to-end delay. The algorithm proposed in [2] was extended in Anjum and Perros [3] to the case where the tagged arrival process is a two-stage Markovian Arrival Process (MAP2). The authors showed experimentally that the MAP2 is a good model for a video trace.

The algorithm described in [3] cannot handle the presence of background traffic that typically competes with a tagged video within the output port of a router. In this paper, we use a CPU-efficient activity-based simulation model to calculate the end-to-end delay of each packet in a tagged video stream in the presence of background video streams, for given packet traces of the tagged and background streams. These traces may represent a single stream or a set of multiplexed streams. Delay percentiles and the jitter can be easily estimated from the calculated end-to-end delay of each packet. In addition, this activity-based simulation model can be trivially extended in order to calculate the packet loss rate. The minimum amount of bandwidth required, so that the three QoS metrics of delay percentile, jitter, and packet loss rate are satisfied, is easily obtained using a simple search algorithm. We show that the bandwidth required for n identical video streams that follow the same path through an IP network, so that that the end-to-end percentile delay remains the same, is a linear function of n. Also, we observed experimentally that for infinite-capacity queues the bandwidth required

to satisfy the percentile end-to-end delay constraint also satisfies the jitter constraint. For finite-capacity queues, the bandwidth required to satisfy both the percentile end-to-end delay and the packet loss rate constraints also satisfies the pair of jitter and packet loss rate constraints.

The paper is organized as follows. In the next section, we describe the tandem queueing network under study and then present the activity-based simulation model. In section 3, we describe the three traces used in our experiments presented in this paper. These traces represent three different video applications, namely, Telepresence, WebEx, and Jabber. In section 4, we consider k identical video streams that follow the same path through an IP network, and we examine the relation between the bandwidth that needs to be allocated on each link along the path as a function of k so that the end-to-end percentile delay remains the same. In section 5, we calculate the required bandwidth so that the constraints on percentile delay and jitter are met. In section 6, we extend the analysis to include the third constraint of the packet loss rate. Finally, the conclusions are presented in section 6.

2. An activity-based simulation model

The tandem queueing network under study consists of N queues, $N \geq 1$, as shown in Figure 1. The tagged arrival process is a video trace that may represent either a single video stream or a multiplexed set of video streams. Each queue in the tandem queueing network is also fed with background traffic that is also described by a video trace (one stream or a set of multiplexed streams). A video trace is a sequence of IP packets identified by the time that the packet arrives and its length in bits. Each queue i has its own local background traffic that does not propagate through one or more queues downstream from i after the packets complete their service at the ith queue. The service time of a packet at each queue i is length/μ_i where "length" is the number of bits in the packet and μ_i is the bandwidth allocated to the ith queue. Each packet in the tagged stream keeps its length throughout the tandem queueing network and all tagged and background packets are served in a FIFO manner in each queue.

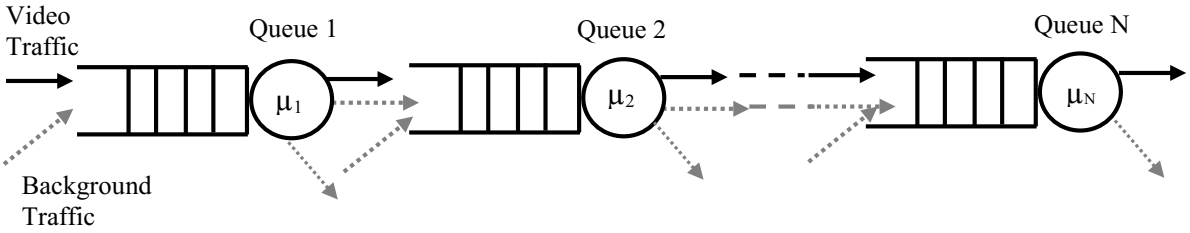


Figure 1: Tandem queueing network under study

In this paper, we present and use an activity-based simulation model as opposed to the commonly used event-based simulation model. In an event-based simulation model, the system under study is associated with a set of events that change the state of the system when they occur. For instance, in a simulation model of a single server M/M/1 queue, an arrival and a

departure are the two events that change the state of the system. i.e., the number of customers in the system. An event-based simulation tracks the occurrence of these events and when an event occurs it takes appropriate action. In an activity-based simulation, the system under study is viewed as a collection of activities or processes. For instance, a single server queueing system can be seen as a collection of the following activities: a) inter arriving, b) being served, and c) waiting for service. In an activity-based design, one mainly concentrates on deriving an algorithm that determine when activities start or stop. Such an algorithm is not always easy to obtain, and in view of this, activity-based simulation models are not very common. For further details, see Perros [17].

Below, we present an algorithm for the activity-based simulation model for calculating the exact end-to-end delay of each packet in a trace. From this, we can obtain any delay percentile and the jitter, where the jitter is defined as the average of the difference of the end-to-end delay of successive packets.

For presentation purposes, we first describe the algorithm for a single queue with no background traffic. Then, we extend it to the case of a tandem network of N queues, $N \geq 1$, with no background traffic, and finally to the case of a tandem network of N queues, $N \geq 1$, with background traffic at each queue. We assume that each queue has an infinite capacity. We then augment the algorithm to cater for finite capacity queues and monitor the number of lost packets to calculate the packet loss rate. This last step is a trivial change to the algorithm, and it is not described in the paper.

Let a packet p arrives at a queue at time $PacketArTime_p$. Let $WaitTime_p$ be the total wait time of the packet p in that queue. This is composed of the time the packet spends waiting in the queue and its service time. The time spent waiting in the queue depends upon whether a packet arrives to find the server empty or not. We will be using the Lindley Equation to calculate the total wait time of the packet. If the server is idle upon arrival of the packet p, then the queueing time is 0. If there is one or more packets in front of p, then its queueing time is the time elapsed from the instance p arrived to the instance that the packet in front of p completes its service. Hence we have the following two cases for the total waiting time:

- If server is free, then $\text{WaitTime}_p = \text{length}_p/\mu$
- If server is busy, then $\text{WaitTime}_p = (\text{ServiceCompletionTime} - \text{PacktArTime}_p) + \text{length}_p/\mu$

where μ bits/sec is the service rate of the queue and *ServiceCompletionTime* marks the time instance when the server becomes free. This variable is updated each time a packet departs the queue.

The algorithm is summarized as follows, where the subscript p refers to the packet that just arrived:

Algorithm 1:

If server is free,
$\text{WaitTime}_p = \text{length}_p/\mu$
If server is busy,
$\text{WaitTime}_p = (\text{ServiceCompletionTime} - \text{PacketArTime}_p)+\text{length}_p/\mu$
$\text{ServiceCompletionTime} = \text{PacketArTime}_p + \text{WaitTime}_p$

The algorithm can be easily extended to multiple queues in a tandem network. For this, we need to make the following changes to the basic algorithm:

1. Each queue i has its associated $\textit{ServiceCompletionTime}_i$ which is the time instance at which the last packet completed its service in queue i, i=1,2,...,N.
2. The sojourn time of a packet is maintained in a cumulative variable TotalWaitTime_p which is obtained by adding the $\text{WaitTime}_{p,i}$ of packet p at each queue i, i=1,2,...,N as it moves through the tandem network.
3. The $\text{PacketArTime}_{p,i}$ of a packet p at queue i is updated as it moves through the network in a way such that the arrival time at queue i would be equal to the departure time from queue $i-1$, i>1. The $\text{PacketArTime}_{p,1}$ is the original arrival time of the packet as given in the trace.

The algorithm is as follows for packet p arriving at queue i:

Algorithm 2:

If server_i is free,
$\text{WaitTime}_{p,i} = \text{length}_p/\mu_i$
If server_i is busy,
$\text{WaitTime}_{p,i} = (\text{ServiceCompletionTime}_i - \text{PacketArTime}_{p,i})+ \text{length}_p/\mu_i$
$\text{ServiceCompletionTime}_i = \text{PacketArTime}_{p,i} + \text{WaitTime}_{p,i}$
$\text{TotalWaitTime}_p = \text{TotalWaitTime}_p + \text{WaitTime}_{p,i}$
If i<N
$\text{PacketArTime}_{p,i+1} = \text{PacketArTime}_{p,i} + \text{WaitTime}_{p,i}$

In summary, the arrival times of the packets from the trace are used to calculate their total waiting time in queue 1 and then their departure times from queue 1. Then the arrival time of each packet to queue 2 is set equal to its departure time from queue 1, and the algorithm repeats. So, if we want to analyze a tandem queueing network of N queues, then we repeat the algorithm N times.

An interesting case arises where all queues have the same service rate, that is, $\mu_i=\mu$, i=1,2,..,N. In this case, the delay incurred by a packet in queue 2 is the same as the delay incurred in queue 3,4,…,N. That is, if the packet delay in a queue i is d_i, then the packet delay

from queue 2 to queue N is $(N\text{-}1)d_2$ and the total delay is $d_1+(N\text{-}1)d_2$. Hence, we only need to run the algorithm for the first two queues.

In order to explain this, we consider two packets, p_1 and p_2, where p_1 is in front of p_2, and we examine the situation at queue 2 at the instance when p_2 has just finished its service at queue 1. So p_2 and arrives at queue 2 and the following two cases are possible:

1. *length of* $p_1 \leq$ *length of* p_2: In this case, p_2 will find no one in queue 2 and will have a zero queueing time, i.e., it will immediately be scheduled for service. This is because by the time p_2 completes its service at queue 1, p_1 would have already completed its service at queue 2 and moved on to queue 3. Not only at queue 2, but p_2 will also have a zero waiting time in all the other queues because each time it reaches a queue p_1 would have already moved on to the next queue.
2. *length of* $p_1 \geq$ *length of* p_2: In this case, p_1 is still in service when p_2 reaches queue 2, and p_2 will have to wait for the remainder of p_1's service, call it r. That is, the queueing time of p_2 at queue 2 is r. This situation repeats when p_2 finishes its service at queue 2 and moves to queue 3, and so on. Packet p_2 will wait r in each of the subsequent queues.

In both cases, the delay that p_2 experiences in queue 2 is the same as the delay it experiences in all the subsequent queues. Hence,

$$\text{TotalWaitTime}_p = \text{WaitTime}_{p,1} + (N\text{-}1)\text{x}\text{WaitTime}_{p,2}$$

Our algorithm for the analysis of the tandem queueing network without background traffic can be easily extended to include background traffic at each queue in the form of a trace. The background traffic maybe a single stream or a superposition of several streams.

The algorithm follows the same basic steps as before. The only difference lies in the selection of the next packet. In the previous algorithm, it was simply the next packet in the trace. However, now we have to make a decision between the next packet of the trace and the next packet of the background trace. This can be resolved by simply comparing the arrival time of the next packet in the tagged trace ($\text{PacketArTime}_\text{p}$) and the next packet in the background trace ($\text{PacketArTime}_{\text{Bkgrd}}$). The algorithm is as follows:

Algorithm 3

For each queue i, $i=1,2,..N$ do:

If ($\text{PacketArTime}_\text{p} < \text{PacketArTime}_{\text{Bkgrd}}$)

If server$_i$ is free,

$\text{WaitTime}_{p,i} = \text{length}_p/\mu_i$

If server$_i$ is busy,

$\text{WaitTime}_{p,i} = (\text{ServiceCompletionTime}_i - \text{PacketArTime}_{p,i}) + \text{length}_p/\mu_i$

ServiceCompletionTime$_i$ = PacketArTime$_{p,i}$ + WaitTime$_{p,i}$
TotalWaitTime$_p$ = TotalWaitTime$_p$ + WaitTime$_{p,i}$
If i<N
PacketArTime$_{p,i+1}$ = PacketArTime$_{p,i}$ + WaitTime$_{p,i}$

Else If (PacketArTime$_p$ > PacketArTime$_{Bkgrd}$)
If server$_i$ is free,
ServiceCompletionTime$_i$ = PacketArTime$_{Bkgrd}$ + length$_{Bkgrd}/\mu_i$
If server$_i$ is busy,
ServiceCompletionTime$_i$ = ServiceCompletionTime$_i$ + length$_{Bkgrd}/\mu_i$

The proposed algorithm calculates the exact end-to-end delay of each packet in the tagged trace, and the packet loss rate for the entire tagged trace. (It also gives exact results for each background process, but these results are of no interest in this study.) Based on the end-to-end packet delay of all the packets in the trace, the mean and any given percentile can be easily computed. In this paper, we use the percentile of the end-to-end delay, as opposed to the mean end-to-end delay, because it is a more useful statistic for SLAs. Specifically, we calculate the 95th percentile, but any other percentile could also be computed. In addition, the algorithm gives the exact jitter, defined as the average of the difference of the end-to-end delay of successive packets. Other metrics of jitter can be easily computed as well.

3. Test Traces

In this paper, we used three different types of traces obtained from Cisco [18]. These traces represent three different video applications, namely, Telepresence, WebEx, and Jabber. We note that the QoS requirements for the end-to-end delay, jitter, and packet loss rate, are stringent for first two applications, and less stringent for Jabber. In this section, we provide some statistics about these three traces.

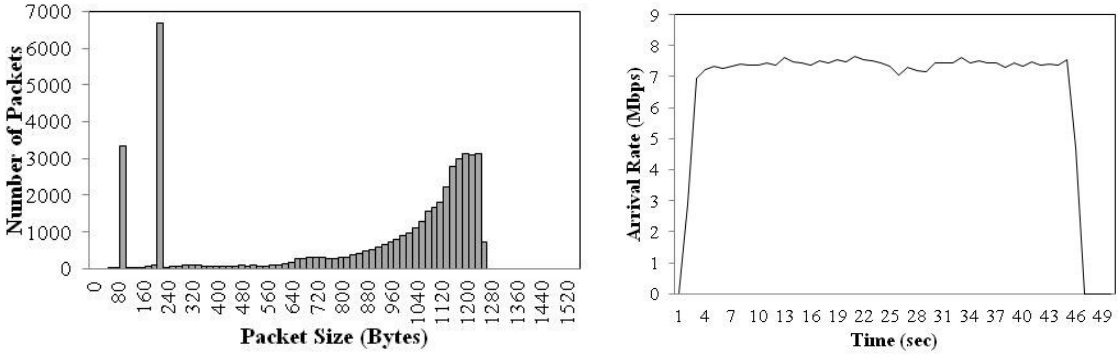


Figure 2: Packet length distribution (left), arrival bit rate (right) for the Telepresence trace.

The Cisco Telepresence trace has a lag-1 autocorrelation (of the successive inter-arrival times) ρ=-0.1383, burstiness (measured by the squared coefficient of variation) c^2=4.40 and an arrival rate λ=7.34 Mbps. The distribution of the packet sizes and the arrival rate (Mbps) are

given in Figure 2. We observe that the packet size varied from 60 bytes to 1260 bytes, and the arrival bit rate is almost constant for the entire trace. The almost constant arrival rate indicates that there are no scene changes in the video, which is normal for teleconferences. In addition to the single Telepresence trace, we will also use different sets of multiplexed Telepresence traces in the paper.

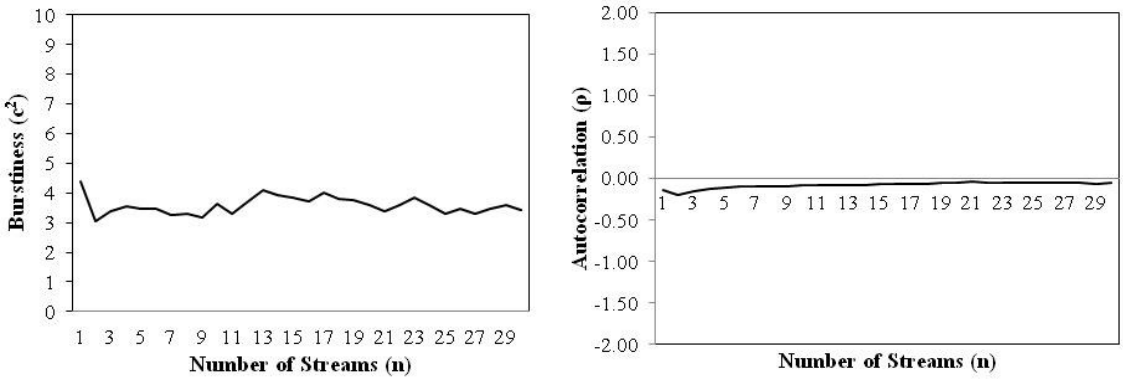


Figure 3: Burstiness (left) and autocorrelation (right) of n multiplexed Telepresence streams

The burstiness c^2 and the autocorrelation lag-1 ρ of n, n=1,2,...,30, combined homogeneous Telepresence streams are shown in Figure 3. We observe that the autocorrelation increases slightly approaching zero. Also, the burstiness remains constant as the number n of streams increases. Typically, it tends to decrease as n increases, but in this case there is no change because the arrival rate of the Telepresence trace is almost constant.

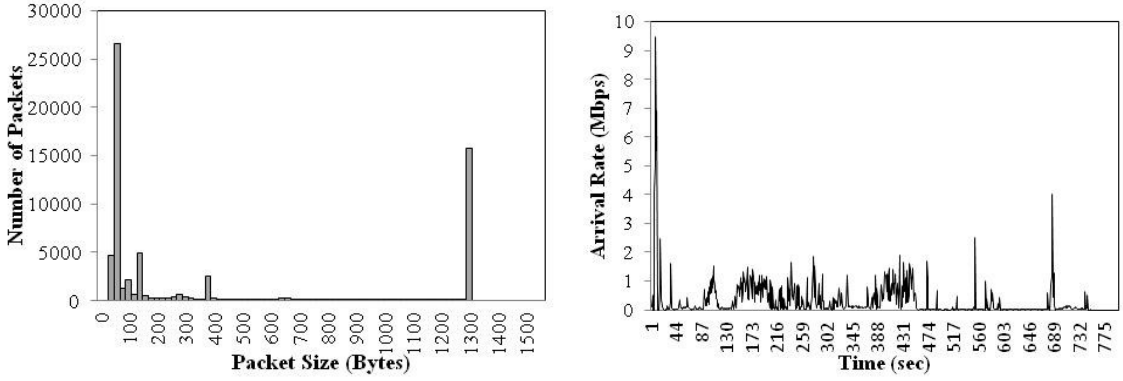


Figure 4: Packet length distribution (left), arrival bit rate (right) for the WebEx trace

The WebEx trace has a lag-1 autocorrelation (of the successive inter-arrival times) ρ=0.0486, c^2=49.5135 and an arrival rate λ=0.309Mbps. The distribution of the packet sizes and the arrival rate (Mbps) are given in Figure 4. The burstiness c^2 and autocorrelation ρ characteristics of n, n=1,2,...,30, combined homogeneous WebEx streams are shown in Figure 5. We note that the autocorrelation increases slightly and the burstiness decreases as the

number of streams increases. The arrival process of the original WebEx trace is highly bursty, and consequently we see a strong decreases of c^2 as the number of streams increases and tends towards a constant value.

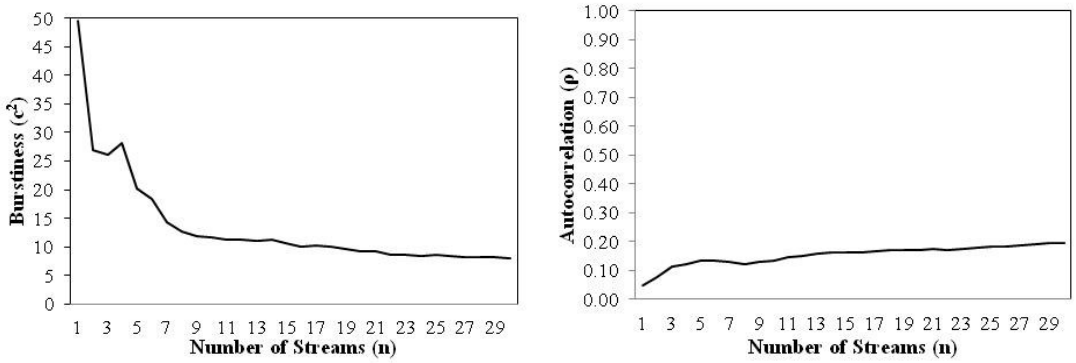


Figure 5: Burstiness (left) and autocorrelation (right) of n multiplexed WebEx streams

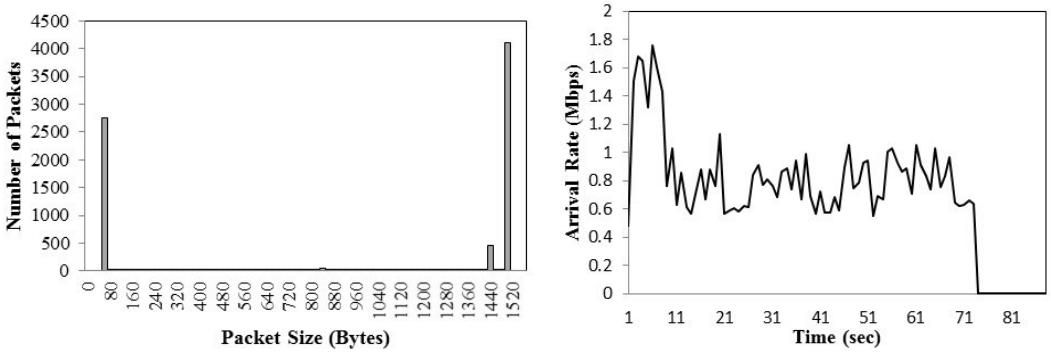


Figure 6: Packet length distribution (left), arrival bit rate (right) for Jabber trace

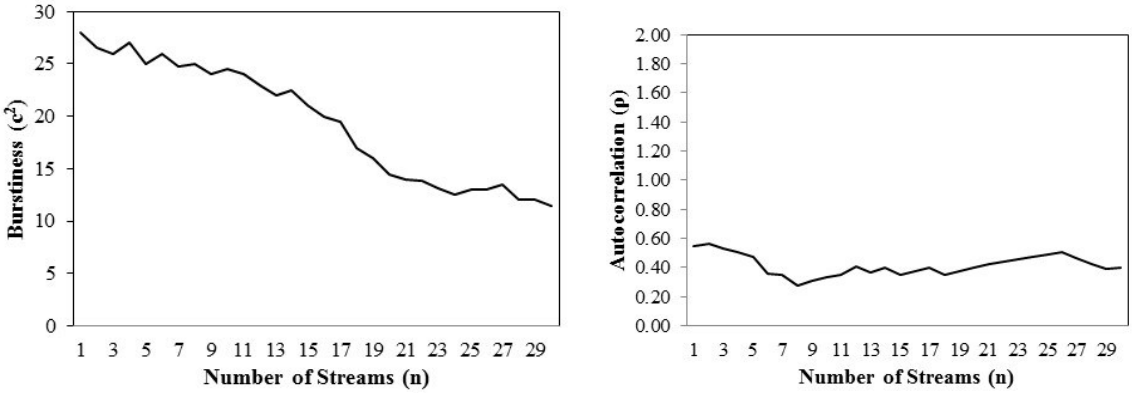


Figure 7: Burstiness (left) and autocorrelation (right) of n multiplexed Jabber streams

Finally, the Cisco Jabber trace has a lag-1 autocorrelation ρ=0.55192, burstiness c^2=28.3756, and an arrival rate λ=0.711 Mbps. The distribution of the packet sizes and the

arrival rate (Mbps) are given in Figure 6. We observe that one fourth of the trace is composed of small signaling packets of size 60 bytes, and the bulk of the traffic is video packets of size 1500 bytes. The burstiness and autocorrelation of n, n=1,2,...,30, homogeneous Jabber streams are shown in Figure 7.

Using the above traces, we timed the execution of the activity-based simulation model and compared it to an event-based simulation model that was developed for comparison purposes. We used a tandem queueing network consisting of 10 queues and we assumed infinite capacity queues, i.e. no packet loss. Each trace was used as the tagged arrival process, and the background traffic consisted of n (n=1, 10, 20, 30) multiplexed streams of the same trace. The execution times (msec) are given in table 1. For each trace, we ran both the activity-based and the event-based simulation models on an Intel i5 processor. The confidence intervals were calculated by replicating the simulation, each time starting from a different state. We ran 30 such repetitions that gave us half the width of the confidence interval within 10% of the estimated 95th percentile of the end-to-end delay. (We note that the same confidence interval was used in all the results presented in this paper, and since it was negligible it was not given in the plots.) We observe that the activity-based model is significantly faster than the event-based simulation, particularly as the number of background traces increases. The activity-based algorithm is of the order O(NP), where N is the number of queues and P are the number of packets. Whereas, the event-based simulation is of the order of O(NP^2).

	Telepresence		WebEx		Jabber	
Background traces	Event-based Simulation	Activity-based Simulation	Event-based Simulation	Activity-based Simulation	Event-based Simulation	Activity-based Simulation
1	1234	246	2414	400	1854	432
10	4293	746	7723	1295	3422	892
20	10036	1656	9625	1374	8722	1032
30	18560	2320	13311	1479	11938	1294

Table 1: CPU time (msec) comparisons: Activity-based vs event-based simulation

4. Bandwidth Requirements for Homogeneous Flows

In this section, we consider n identical video streams that follow the same path through an IP network. That is, they all originate at the same end-point and terminate at the same endpoint. For such a stream of n videos, we examine the relation between the bandwidth that needs to be allocated on each link along the path of the flow as a function of n so that the end-to-end percentile delay remains the same. The results are obtained using the activity-based simulation model for a 10-node tandem queueing network with no background traffic, for the three traces described in above. We assumed that all the queues in the tandem queueing network have the same service rate, since the same bandwidth should be allocated on each link along the path of

the video streams. Consequently, we used the simplified algorithm presented in section 2 where we only have to analyze the first two queues in order to calculate the end-to-end delay. This activity-based simulation model was embedded in a simple search procedure for calculating the bandwidth that should be allocated so that a given 95th percentile is satisfied.

Figure 8, Figure 9 and Figure 10 give results for the Telepresence, WebEx, and Jabber traces respectively for n=1,2,...,30. Each individual trace in the stream was started at a random time uniformly distributed within a time window, so as to avoid temporal synchronization of the traces. The graph showing the required bandwidth is labeled "required bandwidth". In addition, we plotted the average arrival rate of the multiplexed stream, labeled "average bandwidth", and the bandwidth obtained by multiplying the required bandwidth of a single stream times the number of streams, labeled "no statistical gain bandwidth". The latter measure is the bandwidth required assuming that no statistical gain is obtained when multiplexing n streams. Also, the average bandwidth is the least amount of bandwidth required to keep the system stable and is equal to the average arrival rate, which is a linear function of n.

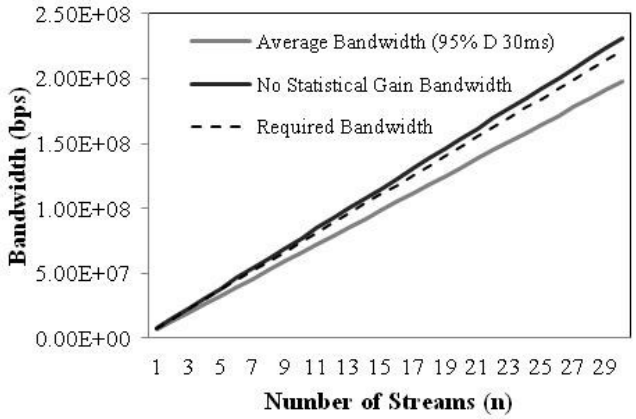


Figure 8: Bandwidth requirement for a 95th percentile delay of 30 msec (Telepresence)

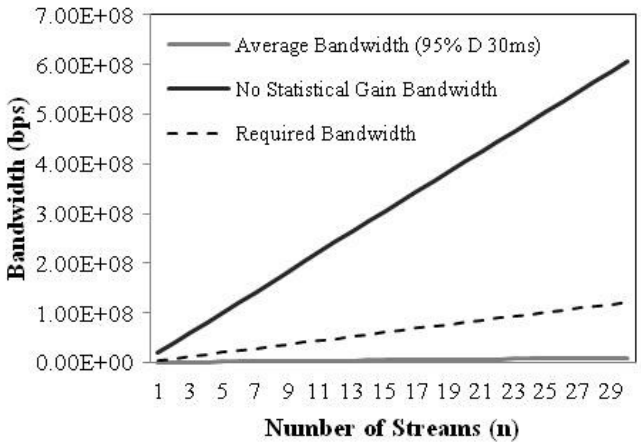


Figure 9: Bandwidth Requirement for Fixed 95th Percentile Delay of 30msec (WebEx)

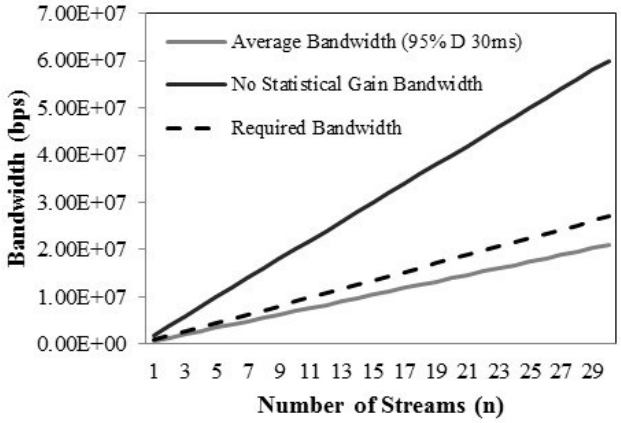


Figure 10: Bandwidth Requirement for Fixed 95th Percentile Delay of 30msec (Jabber)

We note that in the Telepresence case, the three curves are very close to each other, because the transmission rate is almost constant due to lack of scene changes. In the WebEx case, the original trace is very bursty (c^2=49.5135) which means that the statistical gain increases with the number of multiplexed traces, and as a result the required bandwidth and the no statistical gain graphs diverge. In the Jabber case, the graphs behave similarly to the WebEx case. Finally, as the end-to-end percentile delay constraint is relaxed, the required bandwidth graph becomes closer to the average bandwidth graph, as expected (see Anjum [4] for numerical results).

We observe that in all three traces, the required bandwidth is a linear function of n, the number of multiplexed streams. This linearity has also been observed by Lone [14]. This can be explained theoretically using the definition of big-theta Θ that implies asymptotic equality. [6].

5. Bandwidth Estimation under percentile delay and jitter constraints.

In this section we give results on the bandwidth that needs to be allocated on each link along the path of a video flow so that a given percentile of the end-to-end delay and the jitter are both satisfied. We define jitter as the average of the difference of the end-to-end delay of successive packets. In all the experiments, we used the 10-node queueing network with background traffic at each node. In interest of space, we present the results are obtained for the Telepresence and WebEx traces only, using algorithm 3 presented in section 2 in conjunction with the simple search procedure described in the previous section.

We analyzed the 10-node queueing network with background traffic assuming infinite capacity queues. Figure 11 gives the required bandwidth for the Telepresence trace as a function of the background traffic obtained by multiplexing k Telepresence traces, k=1,2,...,30, so that the 95th percentile delay is equal to 50 msec (this reflects only the total queueing delay, and it does not include the propagation delay and the jitter.) The graph for the jitter shown in the same figure, gives the resulting jitter values for the selected bandwidth that satisfied the percentile delay. Figure 12 gives the required bandwidth for the Telepresence trace as a function of k, k=1,2,...,30, so that the jitter is equal to 30 msec. The graph for the percentile delay shown in the same figure, gives the resulting percentile delay values for the selected bandwidth that satisfied the jitter constraint.

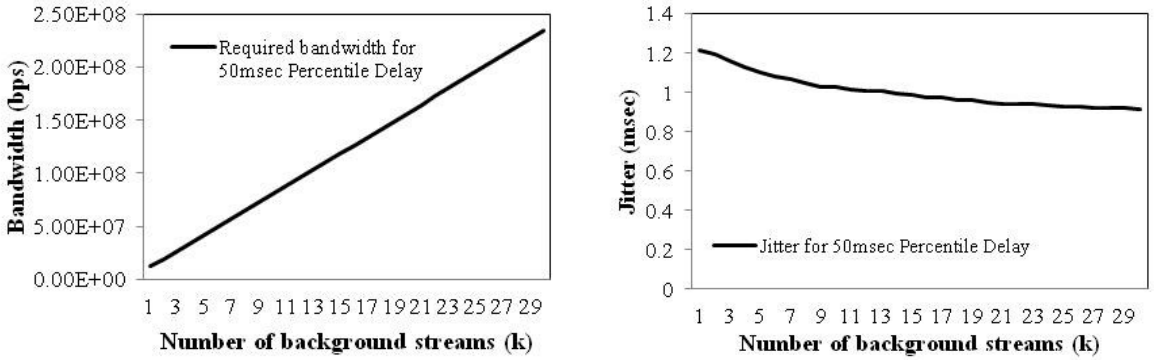


Figure 11: Required bandwidth and jitter values for a 95th percentile delay of 50msec (Telepresence)

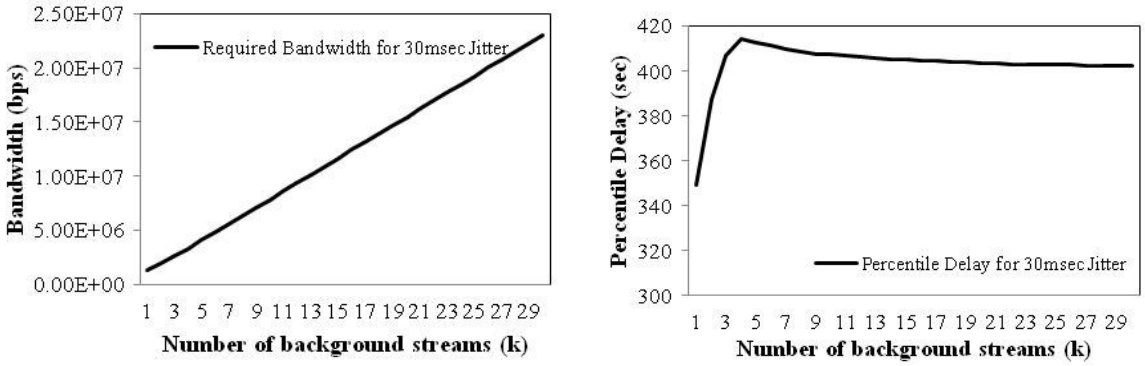


Figure 12: Required bandwidth and percentile delay values for jitter of 30msec (Telepresence)

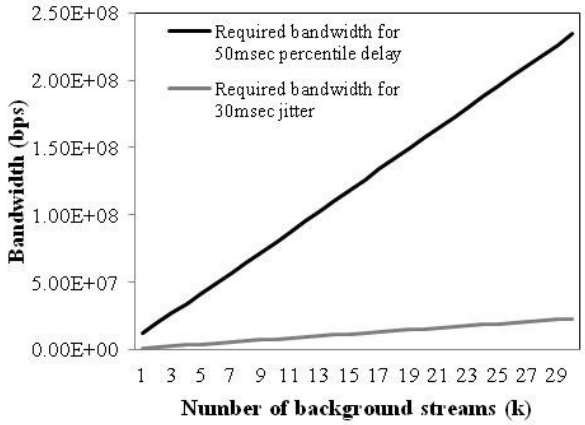


Figure 13: Required bandwidth for each constraint (Telepresence)

The bandwidth that satisfies both constraints was calculated iteratively. That is, for a given k we calculate the end-to-end percentile delay and jitter values assuming an initial small value for the bandwidth, so that neither constraint is met. Next we increase the bandwidth value by a fixed step size, and re-calculate the end-to-end percentile delay and jitter. After a few iterations the less stringent constraint, which in our experiments is always the jitter constraint, will be met. We store this bandwidth value and keep on iterating until the second constraint, which is the percentile end-to-end delay, is also satisfied. This is the required bandwidth that satisfies both constraints.

The results for k=1,2,...,30, obtained are presented in Figure 13. The grey curve indicates the bandwidth required to satisfy the jitter value of 30 msec, and the black curve represents the bandwidth required to satisfy the 95th percentile of the end-to-end delay of 50 msec. As can be seen, the required bandwidth for the percentile delay constraint also satisfies the jitter constraint.

We repeated the same experiments as above for the WebEx trace. The required bandwidth that satisfies each constraint separately is given in Figure 14. Again we observe that the required bandwidth for the percentile delay constraint satisfies the jitter constraint, and hence it is the bandwidth that should be allocated on each link.

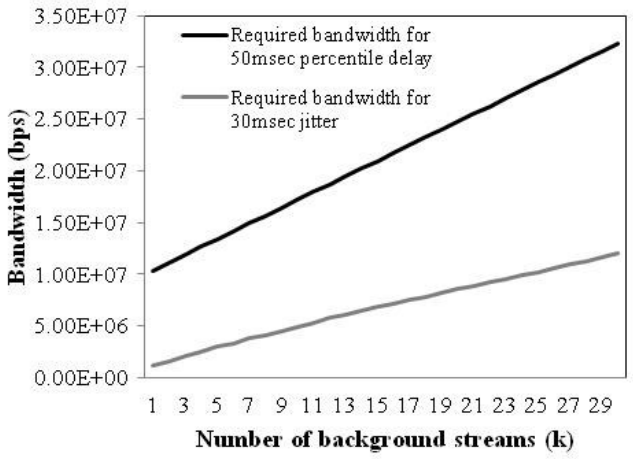


Figure 14: Required bandwidth for each constraint (WebEx)

In the above numerical results we see that the delay constraint dominates the jitter constraint for values of 50 msec and 30 msec respectively. This dominance was confirmed for the two traces for different values of the end-to-end delay and jitter.

In order to confirm the generality of this result, we repeated the above experiment with different traces. We summarize our findings at the end of section 5.

5.1 Generalization using MAP2 Arrival Process In order to confirm the generality of this result, we repeated the above experiment with different traces. These traces were generated from a generalized 2-stage Markovian Arrival Process (MAP2). (As indicated in the Introduction, the MAP2 has been shown in [3] to be a good model for a video packet flow.) Using a theoretical model to generate traces permited us to vary the lag-1 autocorrelation ρ and burstiness c^2 of a trace. The traces were generated by first setting the parameters of the MAP2 so that the arrival process corresponds to a given ρ and c^2, and then generated the inter-arrival times of the packets by simulating the MAP2 The required confidence intervals were calculated using the batch means method. The length of each packet in the trace was obtained by sampling from the packet-length distribution of the Telepresence trace given in Figure 2. Below, we briefly review the MAP2 and then give the results.

A MAP is a process that counts transitions of a finite continuous-time Markov chain with m states. The size m is called the order of the MAP, and determines the dimensions of matrices D_0 and D_1:

$$D_0 = \begin{pmatrix} -q_{11} & q_{12} & \cdots & q_{1m} \\ q_{21} & -q_{22} & \cdots & q_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ q_{m1} & q_{m2} & \cdots & -q_{mm} \end{pmatrix}, D_1 = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mm} \end{pmatrix}$$

where $q_{ii} = \sum_{j=1, j\neq i}^{m} q_{ij} + \sum_{j=1}^{m} a_{ij}$. D_0 and D_1 represent the transition rates of the MAP process and define the infinitesimal generator $D = D_0 + D_1$, see [15] and [16]. The MAP2 process is uniquely defined by the following six parameters: q_{12}, q_{21}, a_{11}, a_{12}, a_{21}, a_{22}, with $q_{11} = q_{12} + a_{11} + a_{12}$ and $q_{22} = q_{21} + a_{21} + a_{22}$. The steady state probability vector π for a MAP2 process is defined as follows [12]:

$$\pi(1,1) = \frac{a_{21} + q_{21}}{a_{21} + q_{21} + a_{12} + q_{12}}, \pi(1,2) = \frac{a_{12} + q_{12}}{a_{21} + q_{21} + a_{12} + q_{12}}$$

The average rate of arrivals in a MAP is called the fundamental rate of the MAP. It is given by $\lambda = \pi D_1 e$, where π is the stationary probability vector in the Markov chain (i.e., $\pi D = 0$ and $\pi e = 1$). The marginal distribution of the inter-arrival time of the MAP is a phase-type distribution PH(α,D_0), where $\alpha = \pi D_1 / \lambda$, which is the stationary probability vector. Since the marginal distribution of the inter-arrival time in a MAP is a PH distribution, its moments, density and distribution function can be calculated using the standard formulae.

For the interval stationary MAP, we have the following inter-arrival time distribution function $F(x) = Pr\{X \leq x\}$:

$$F(x) = \alpha(I - e^{D_0 x}(-D_0)^{-1} D_0 e_1$$

and the following moments of the inter-arrival time X:

$$m_z \equiv E\{X^z\} = z!\, \alpha(-D_0)^{-(z+1)} D_1 e, z = 1,2, \ldots$$

From the joint distribution function, we have the autocovariance function $\psi[z]$ ($z \geq 1$) of the inter-arrival times:

$$\psi[z] \equiv E[\{X_i - m_1\}\{X_{i+z} - m_1\}] = \alpha {D_0}^{-2} D_1 [T^{z-1} - ep] {D_0}^{-2} D_1 e$$

The results are presented in a series of graphs given in Figure 15 for the 10-node tandem queueing network. For each graph we plot a curve indicating the bandwidth required to satisfy the jitter value of 30 msec, and a curve indicating the bandwidth required to satisfy the 95th percentile of the end-to-end delay of 50 msec. Each figure corresponds to a MAP2 with different autocorrelation lag-1 ρ and c^2. The values for ρ and c^2 were obtained from the set of all feasible values for ρ and c^2 for a MAP2. The lag-1 autocorrelation ρ for a MAP2 varies from -0.5 to 0.5. As can be seen, the required bandwidth for the delay constraint completely dominates that for the jitter constraint and hence it is the bandwidth that satisfies both the constraints.

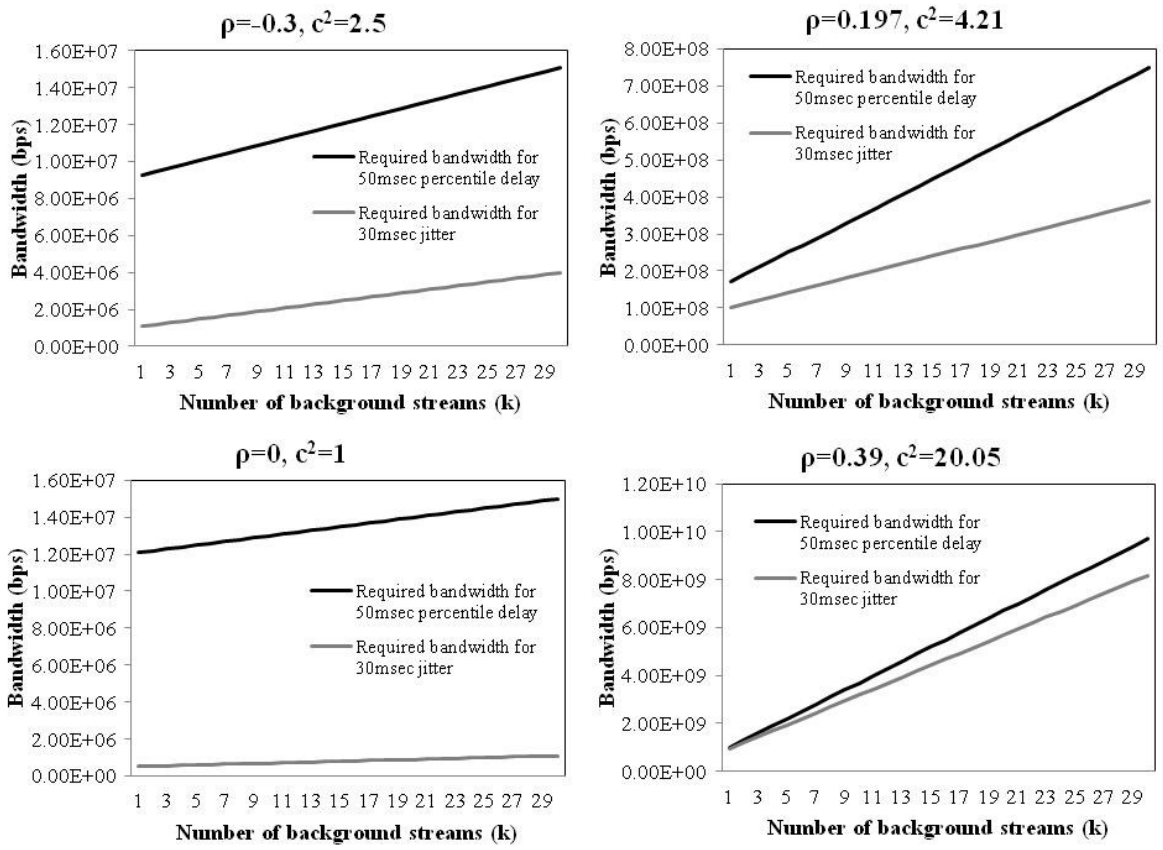


Figure 15: Required bandwidth for each constraint

We note that the required bandwidth for both percentile delay and jitter constraints, become steeper as ρ and c^2 increase. Also, the required bandwidth for the jitter constraint becomes closer to the required bandwidth for the percentile delay constraint as the burstiness increases. This makes sense intuitively as higher level of burstiness translates into larger jitter values for the traffic. However, the percentile delay constraint of 50 msec still dominates the jitter constraint of 30 msec.

Obviously if the delay constraint is relaxed, it is possible that the jitter constraint may dominate. We determined the cross over point, i.e., the point where the jitter constraint of 30 msec starts dominating the percentile delay constraint, for different levels of burstiness. Specifically, we varied c^2 from 0 to 30, kept the jitter constraint fixed at 30 msec, and determined the value of the percentile delay constraint where it loses its dominance over the jitter constraint. The results are presented in Figure 16.

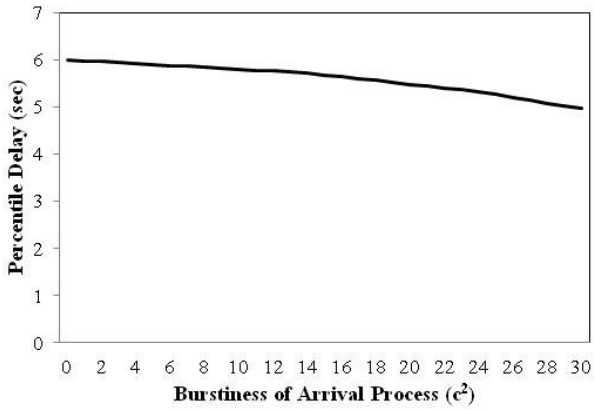


Figure 16: 95th Percentile Delay values for which jitter constraint of 30msec starts dominating

The results derived from this section can be summarized as below:

1. The dominance of the delay constraint is maintained even when the constraint is relaxed from the order of milliseconds to the order of a few seconds.
2. Even for high burstiness values, the jitter constraint of 30 msec does not start dominating until the delay constraint is relaxed beyond 4 sec. Such high values of delay are quite unrealistic for video based (and audio based) real-time multimedia services.
3. Hence, we can safely conclude that for bandwidth estimation of video traffic, the percentile delay bound dominates the jitter bound.

6. Bandwidth estimation under percentile delay, average jitter and packet loss rate constraints

In this section, we extend our analysis to include the packet loss rate as an additional constraint. The results were obtained only for the Telepresence and WebEx traces. Algorithm 3 was modified to account for finite capacity queues. (The modification is trivial and it is not reported in this paper.) We assume that each queue in the 10-node queueing network has the same finite capacity and the same service rate.

It is important to note that for finite queues we cannot study the end-to-end delay, jitter and packet loss independently. Let us consider the packet loss and the percentile delay constraints together for a finite buffer K. Let us assume that the value of K has been fixed such that it results to a packet loss of $x\%$ for a given bandwidth value of μ. Let us also assume that for this value of μ, the corresponding value of the end-to-end percentile delay is D. The interesting question is that if we increase the value of μ to μ', will the corresponding end-to-end delay value D' be greater than or less than D? Certainly an increase in bandwidth reduces the end-to-end delay, so we are tempted to say that $D' < D$. However, at the same time, an

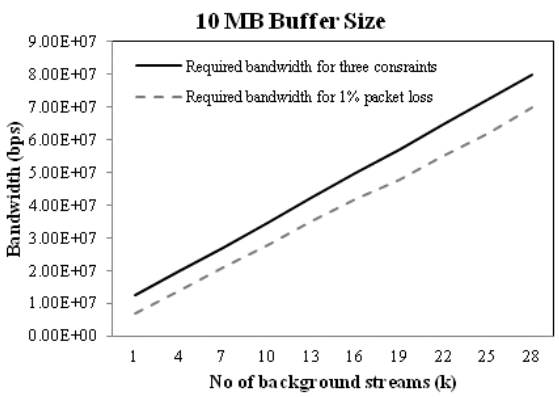
increase in the bandwidth also reduces the packet loss, which means that more packets are now on their way from source to destination, which may increase congestion and in turn may increase the end-to-end delay hence resulting in $D' > D$!

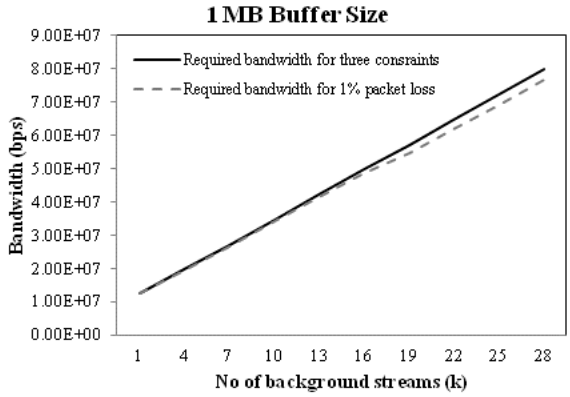
Similar arguments can be applied to the jitter and the packet loss constraints. Thus increasing the bandwidth without fixing the packet loss results in competing conditions for both increasing and decreasing the end-to-end delay/jitter and the relationship becomes complex to understand.

To get a clearer picture of how the three constraints affect the bandwidth requirements of a traffic flow, we classify them to one primary and two secondary constraints. The primary constraint is the packet loss rate that should be less than 1%. The first secondary constraint is that the 95th percentile of the end-to-end delay should be less than 50 msec and the second secondary constraint is that the jitter should be less than 30 msec. In our analysis, the primary constraint will always be met and we will look at it in isolation and also in combination with one or both of the secondary constraints.

We focus on the following two questions. First, how does the bandwidth required to satisfy the primary constraint compare to the bandwidth required to satisfy both the primary and the two secondary constraints? That is, does satisfying the packet loss rate constraint also satisfy the end-to-end percentile delay and jitter constraints? Second, which (primary, secondary) pair of constraints dominate the other in terms of bandwidth requirement? In other words, does the bandwidth required to satisfy the packet loss and the end-to-end delay constraints dominate the bandwidth required to satisfy the packet loss and the jitter constraints, or vice versa?

The iterative scheme was run for three different buffer sizes, 10 MB, 1 MB and 50 KB, and for k=1,2,...,30. The three buffer sizes result in approximately 10%, 25% and 80% buffer utilization respectively for the Telepresence trace, and in approximately 8%, 20%, and 80% buffer utilization respectively for the WebEx trace.





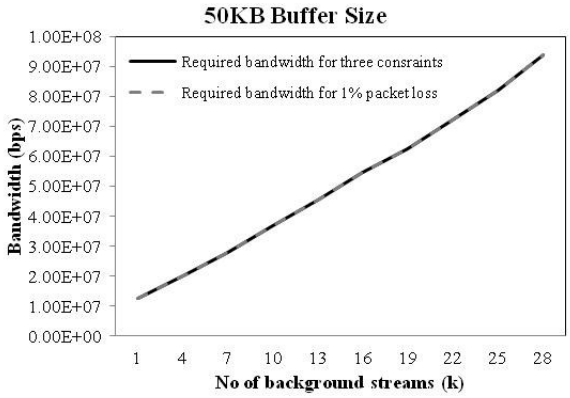
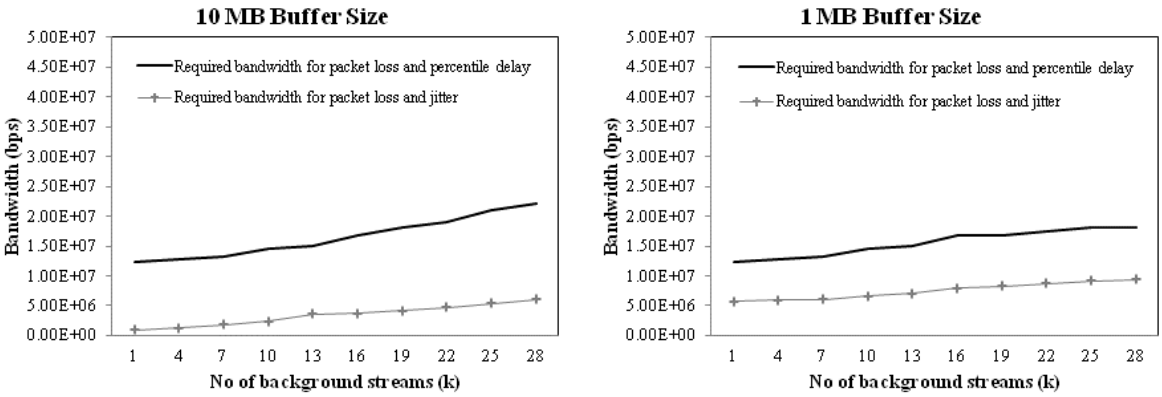


Figure 17: Required bandwidth that satisfies the constraints (Telepresence)

Figure 17 gives results for the Telepresence trace. The dotted line indicates the required bandwidth that satisfies the primary constraint (packet loss rate = 1%), the solid black line indicates the required bandwidth that satisfies both the primary and the two secondary constraints. We see that the bandwidth required to satisfy the three constraints dominates the bandwidth required to satisfy the packet loss rate only, and the difference increases as the buffer size increases. Similar results were obtained for WebEx trace which we are excluding in favor of the length of the paper.

Let us shift our focus on the second question, i.e., which (primary, secondary) pair dominates the other in terms of bandwidth requirement. The results obtained for the WebEx are given in Figure 18. We notice that for large buffer sizes, the bandwidth required to satisfy the packet loss and percentile delay constraints dominates the bandwidth required to satisfy the other pair of constraints, namely, the packet loss and jitter constraints. Similar trends are observed for the Telepresence trace also which are omitted here. For the case where the buffer size is 50K, the bandwidth is the same for both pairs of constraints.



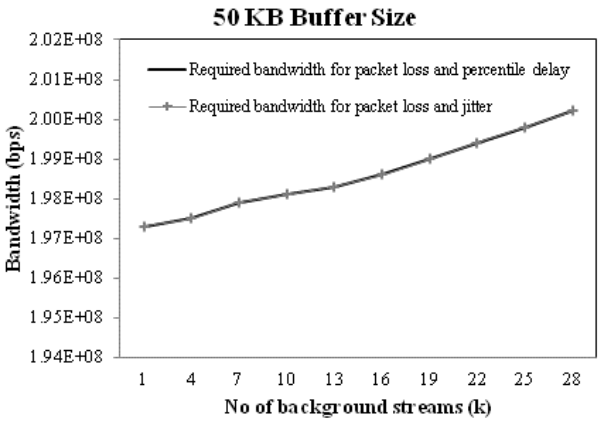


Figure 18: Required bandwidth that satisfies constraints (WebEx)

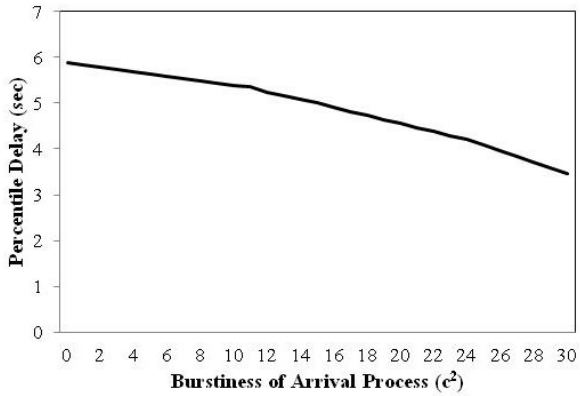


Figure 19: 95th Percentile Delay values for which the pair of constraints (packet loss rate=0.01, jitter=30 msec) starts dominating

We explored the generality of these conclusions by carrying out similar experiments as in the previous section using the MAP2 process. Figure 19 gives the cross over point at which the pair of constraints (packet loss rate=0.01, jitter=30 msec) dominates the pair of constraints (packet loss rate=0.01, end-to-end delay) for different values of c^2. We note that this dominance holds for very high and unrealistic percentile delay values, which is in seconds.

7. Conclusions

In this paper we presented and used a CPU-efficient activity-based simulation design for calculating the sojourn time of a packet and the packet loss rate in a tandem queueing network depicting the path of a video flow, which is characterized by a packet trace. Background traffic, also characterized by a trace, is allowed in the queueing network. From the sojourn time we can easily calculate any given percentile of the end-to-end delay and the jitter. The required bandwidth that satisfies all three constraints is easily obtained using a simple search algorithm.

In our analysis we used real traces and also generalized our results using traces generated by a theoretical model of a video arrival process depicted by a Markovian Arrival Process. We showed that the bandwidth required for n identical video streams that follow the same path through an IP network, so that that the end-to-end percentile delay remains the same, is a linear function of n. We also observed experimentally that for infinite-capacity queues the bandwidth required to satisfy the percentile end-to-end delay constraint also satisfies the jitter constraint. For finite-capacity queues, the bandwidth required to satisfy both the percentile end-to-end delay and the packet loss rate constraints also satisfies the pair of jitter and packet loss rate constraints.

References

[1] Quality of Service Design, Cisco Whitepaper

[2] B. Anjum, H. Perros, X. Mountrouidou, and K. Kontovasilis, "Bandwidth allocation under end-to-end percentile delay bounds", International Journal of Network Management, Wiley, vol. 21, no. 6, pp. 536-547, 2011.

[3] B. Anjum and H. Perros, "An approximation of the percentile of the end-to-end delay for MAP2 arrivals with an application to video traces", Networks 2012.

[4] B. Anjum, "Bandwidth allocation under end-to-end percentile delay bounds", PhD Thesis, North Carolina State University, 2012.

[5] M. Conti, E. Gregori, and I. Stavrakakis, "Large impact of temporal/spatial correlations on per-session performance measures: single and multiple node cases", Performance Evaluation, vol. 41, pp. 83-116, 2000.

[6] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, Introduction to Algorithms, 3rd ed. Massachusetts Institute of Technology, 2009.

[7] S. Dasu, "Class dependent departure process from multiclass phase queues: Exact and approximate analyses", European Journal of Operational Research, vol. 108, no. 2, pp. 379-404, 1998.

[8] H. W. Feng and J. F. Chang, "Connection-wise end-to-end performance analysis of queueing networks with MMPP inputs", Performance Evaluation Journal, vol. 43, pp. 39–62, 2001.

[9] G. Geleji and H. Perros, "Jitter analysis of an IPP tagged traffic stream in an {IPP,M}/M/1 queue", to appear in the Annals of Telecommunications.

[10] G. Geleji and H. Perros, "Jitter analysis of an MMPP-2 tagged stream in a two-class {MMPP-2, MMPP-2}/M/1 queue and a tandem queuing network consisting of similar queues", to appear in Applied Mathematical Modeling.

[11] R. L. Ioannis and I. Stavrakakis, "Traffic shaping of a tagged stream in an ATM network: approximate end-to-end analysis", in IEEE INFOCOM, 1995, pp. 162-169.

[12] S.H. Kang, Y.H. Kim, D.K. Sung and B.D. Choi, "An application of Markovian arrival process (MAP) to modeling superposed ATM cell streams," IEEE Transactions on Communications, vol. 50, no. 4, pp. 633-642, 2002.

[13] J. Kumaran, K. Mitchell, and A. van de Liefvoort, “An analytic model of correlations induced in a packet stream by background traffic in IP access networks”, in Proceedings of the 19th International Teletraffic Congress, Beijing, China, 2005, pp. 687-696.

[14] Q. Lone, “Bandwidth allocation for video streams subject to an end-to-end percentile delay”, MS Thesis, North Carolina State University, 2011.

[15] D. M. Lucantoni, K. S. Meier-Hellstern, and M. F. Neuts, "A single-server queue with server vacations and a class of non-renewal arrival processes," Advances in Applied Probability, vol. 22, no. 3, pp. 676-705, 1990.

[16] D. M. Lucantoni, "New results on the single server queue with a batch Markovian arrival process," Communications in Statistics - Stochastic Models, vol. 7, pp. 1-46, 1991.

[17] H. Perros, *Computer simulation techniques - the definitive introduction!*, http://www4.ncsu.edu/~hp//simulation.pdf.

[18] V. Puttasubappa, Private Communication.