Les Cahiers du GERAD ISSN: 0711–2440

End-to-end network queuing model equivalent for video application

H. Dbira, A. Girard,
B. Sansò

G–2016–85

November 2016

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

End-to-end network queuing model equivalent for video application

Hadhami Dbira [a]
André Girard [b]
Brunilde Sansò [a]

[a] *GERAD & Electrical Engineering Department, Polytechnique Montréal (Québec) Canada, H3C 3A7*

[b] *GERAD & INRS-EMT & Electrical Engineering Department, Polytechnique Montréal (Québec) Canada, H3C 3A7*

hadhami.dbira@polymtl.ca
andre.girard@gerad.ca
brunilde.sanso@polymtl.ca

November 2016

Les Cahiers du GERAD
G–2016–85

Abstract: Network characterization and modelling is an important issue to understand and monitor IP network performance, in particular for real-time multimedia applications. To maintain an adequate quality of experience for end customers, we need to monitor and eventually control the effects of the network behavior. Thus, an accurate network performance model is needed, which is not a simple task, given the complex dynamics of a network. In this paper, we propose to represent the effect of an IP network on a video connection as a single G/M/1 queue. When a video session is set up, the two ends can infer the queue parameters based only on local measurements of the stream itself and with a single exchange of information between the two ends. We found that the recommended queue parameters may be different from one connection to the next but that the actual shape of the arrival process does not have a large impact on the selected parameters. In fact, using a lognormal distribution may work for many cases. Finally, we found that jitter can be used as a proxy to estimate network delay without the need for end-to-end synchronization.

Keywords: G/M/1 queue, video applications, end-to-end jitter, TCP/UDP

Acknowledgments: This work was supported by Grant No. 121949-2012 from the Natural Sciences and Engineering Research Council of Canada.

1 Introduction

Multimedia services over IP (Internet Protocol), particularly video applications, have been rapidly increasing. Cisco [1] predicts that by 2019, video users will generate around 80 to 90 percent of the total traffic. Content providers and network operators are increasingly competing to offer high-quality digital streamed and real-time video services both to wired and wireless users. A wide range of video streaming applications, such as video on demand (VoD) or peer-to-peer video streams (P2P), are carried by the Transmission Control Protocol (TCP). On the other hand, real-time video services, such as video conferencing, tele-medicine and live streaming often use the User Datagram Protocol (UDP) or Real-Time Protocol (RTP) over UDP.

Delivering this kind of applications through IP networks raises some challenging issues. A good quality of experience (QoE) for the user can be achieved only through strict quality of service (QoS) requirements in terms of bandwidth, end-to-end transit delay, end-to-end jitter and packet loss. Video traffic is particularly sensitive to end-to-end jitter, that can cause some undesirable effects such as pixelation and frozen images. A number of techniques such as video compression and playback buffering mechanisms are concurrently used to manage the QoS and offer a suitable video quality.

Simple QoS management is based on measuring and collecting some performance metrics but this falls short of providing simple models to understand and recreate the behavior of the network. In our view, in order to be more effective in controlling QoS, simple and accurate modelling of the network behavior is needed. In this paper, we propose that the effect of the whole IP network on a video connection be represented by a single G/M/1 queue whose parameters are estimated through very simple measures at the two ends and a single exchange of information.

1.1 Related work

The traditional approach for network planning and control is to model the network as a queuing system and derive from this the appropriate QoS metrics from which one can extract the relevant QoE values. It is possible to get a full characterization of network performance if the network is modelled as a network of M/M/1 or M/M/1/K queues [2, 3, 4, 5]. One can get simple solutions because of the simplicity of the Poisson model. It provides a simple model for end-to-end delay, packet loss and jitter estimation, which can be used in network planning. Authors in [6] conclude that there are cases where it is possible to estimate end-to-end delay in core networks by modelling it with a series of M/D/1 queues.

However, there is a large body of work [7, 8] showing that multimedia traffic, and in particular video traffic, shows a long-range dependence so that the Poisson model is definitely not an accurate description of link traffic. Extending the classical techniques to non-Poisson traffic is much more difficult.

An estimation technique for the end-to-end delay is described in [9]. This is based on measuring the queue length at the nodes on a path. The authors assume that the input process is Gaussian, which includes a large class of self-similar process. They fit the measured distribution with the asymptotic queue length and reconstruct from this the sojourn time in the queue. The end-to-end delay is estimated by convolution.

The work of [10] gives formulas for the jitter in case of On-OFF traffic like VoIP applications with constant service time. This is based on the assumption of independent and exponential transit times for consecutive packets. This is extended in [11] to the case of a network path using the techniques first proposed in [4].

Instead of trying to derive the QoS parameters of a session from the network links, a different approach is to monitor the session itself and to infer from this the relevant QoS parameters. Most of these techniques are based on the assumption that the behavior of the session is determined by a single bottleneck link where congestion occurs. Many techniques, such as the packet-pair method, also inject artificial traffic on the session to estimate its performance. See [12] for a discussion of these techniques.

Other authors, as in [13, 14], try to derive an equivalent queue model based on a single bottleneck in a M/M/1/K or M/D/1/K queue. They use standard queuing theory to extract some performance metrics. For the M/M/1/K queue, they can get a large number of metrics such as packet loss, bandwidth capacity and the background traffic intensity on the link. This is much more difficult for the M/D/1/K where only

a few metrics can be derived, and with considerable numerical effort. These are then compared to actual end-to-end network measurements to determine the accuracy of the model. They conclude that M/M/1/K queue gives a reasonable results for estimating these metrics.

1.2 Contribution

The main contribution of this paper is to propose an equivalent G/M/1 queue to model network performance that can estimate fairly accurately the network transit time and jitter using only very simple local measurements. We use jitter as a proxy to estimate those measurements without the need for end-to-end synchronization. This is based on some previous results on network jitter evaluation, combined with the analysis of traffic traces for TCP and UDP traffic. Differently from [13] and [14], we don't assume a bottleneck link and don't use artificial traffic either. The constraint is that only measurements that are locally available can be used so that the clocks at the two ends need not be synchronized. The tradeoff is between the amount of information measured and transferred between the users on the one hand and the assumptions that are needed to arrive at a queuing model on the other hand.

There are some advantages in having a credible queuing model for the network. First, we can compute the potential impact that a change in the source parameters could have on some performance metrics such as delay and jitter. Conversely, a queuing model can help us decide whether an observed change in some performance measure is simply due to the stochastic nature of the queue or whether network conditions have changed. In that case, it is possible to express this as a change in the parameters of the queue and use this to control the source or the receiver rates.

1.3 Paper structure

This process can be summarized as follows. First, we propose in Section 2 a G/M/1/ queue with infinite buffer and First-Come First-Served (FCFS) service and discuss some assumptions related to this queue.

Next, we consider in Section 3 streaming video over TCP. We propose an algorithm where the two ends of the session can make measurements of the session traffic and from this, infer a *"best"* equivalent queue.

We then briefly discuss in Section 5 how this algorithm could be used for real-time traffic carried over UDP. We mention that the algorithm cannot be used directly with raw UDP but that using RTP would provide enough information to identify the equivalent queue.

Finally, we discuss in Section 6 the relatively small impact of the shape of the distribution of the inter-arrival time on jitter and show that it depends mostly on first two moments of the distribution.

2 The G/M/1 queue

In order to identify a queue, we need four elements:

1. The service discipline. Here, we assume FIFO
2. The buffer size.
3. The arrival process
4. The service process

First, we assume that the queue equivalent to the whole network path, both core and access, is a G/M/1 queue, as shown in Figure 1. This model has two important features. One is that given the arrival process G and the sojourn time distribution, we can compute the service rate exactly. The other is that given this service rate, one can compute the mean jitter analytically. We think that this assumption is acceptable for the following reasons.

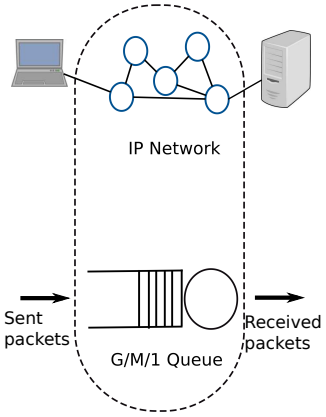

Figure 1: Network G/M/1 queue equivalent

2.1 Infinite buffer

We assume that we have no losses so that we can assume that the buffer is infinite. This assumption is based on the measurements made on some real video sessions where we find that the packet loss is in fact very small and can be neglected. Note that for a real-time service, a packet that is delayed more than a certain time, either because of network congestion or retransmission, is considered lost since it cannot be used to reconstruct the video frame once this frame has been sent to the application.

2.2 Generic arrival process

There is clearly an advantage in modeling the traffic entering the network as an arbitrary process with a general distribution for the packet inter-arrival time. This avoids the Poisson assumption which is not considered realistic for multimedia traffic [7, 15, 8], at least in the core.

The discussion so far has assumed that we know the complete inter-arrival time distribution f_R at the source. In practice, this may not be known and we may have to measure the process to get an estimate of the distribution. This is complicated and may take some time to get reliable statistics. To simplify the procedure, we make a further assumption that *The source inter-arrival time can be modelled by a two-parameter distribution.* This is a relatively strong assumption but as we will see, it turns out to be reasonably accurate for the sources we have considered here.

2.3 Exponential service time

A much stronger assumption is that of an exponential service time. We think that this is not unrealistic for the following reasons.

First note that the service time in question is *not* the service time at a particular network queue. Rather, it is an aggregated measure of the time spent in the network: It is the sum of all the waiting and service times experienced by a packet as it crosses the network. The reason why a Poisson model may be accurate is that on each link, the waiting time of a packet depends on all the cross-traffic that share the queue of the packet under consideration. Two consecutive packets of the same stream may be separated by a number of packets from other streams. To the extent that these background streams are all different and uncorrelated, it is not unreasonable to assume that the total time in the queue for one packet is in effect driven by a Poisson process.

Another reason why the exponential assumption may be realistic is based on the work of [14] where an the exponential service time seems to be quite a good approximation for end-to-end service time through the network. Finally, we have made some measurements on some real streaming video flows carried over TCP. We have measured the Round-Trip Time (RTT) using the ACK packets from the client to the server

and plotted the corresponding histogram with a fitted exponential. Some of our results are shown on Figures 2 and 3. We can see that the agreement is quite good except at low values of the transit delay where the exponential distribution is clearly over-estimating the frequency of short delays. Based on this and other similar results not presented here to conserve space, we use the Poisson service time model in the following.

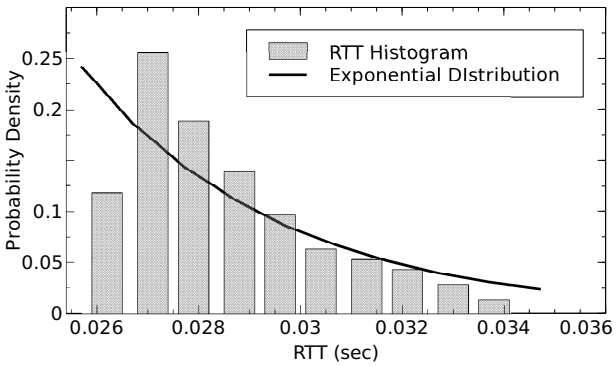

Figure 2: Approximate packet transit time vs exponential distribution- V-2 experiments

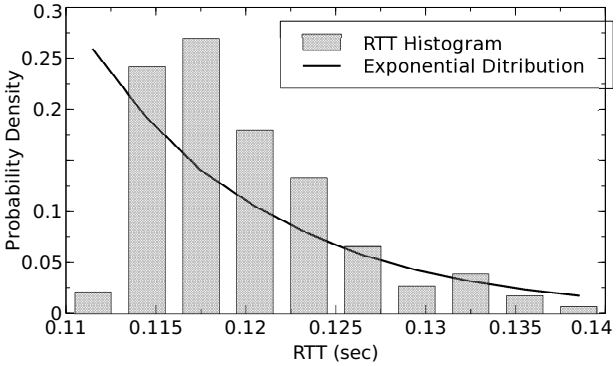

Figure 3: Approximate packet transit time vs exponential distribution- V-5 experiments

3 Estimation algorithm for streaming over TCP

The question that we examine in this section is how the two ends of the connection can use this information to arrive at a consistent estimation of the queue parameters.

At first, we examine a relatively frequent case where video is streamed over a TCP connection. The traffic from the source to the client is made up mostly of data packets while the traffic in the other direction is made up mostly of acknowledgement (ACK) packets from the client to the server.

3.1 Estimation procedure

We present the estimation algorithm where the computation is made at the source but it could be obviously done at the receiver as well. In all that follows, we try to keep the measurements as few and as simple as possible and to limit to the maximum the amount of information that has to be transmitted between the source and the destination.

First, we define variables that are used by this algorithm

J_m	Measured Jitter
t_i	time at which the packet i is sent by the source
r_i	time at which the packet i is received at the destination
R_i	inter-arrival time, $R_i = t_i - t_{i-1}$
O_i	inter-departure time, $O_i = r_i - r_{i-1}$
f_R	probability density function (pdf) of inter-arrival time variable R
f_R^r, $r \in \{l, g, p\}$	log-normal pdf, gamma pdf, Pareto Type I pdf
$\hat{J}^r$, $r \in \{l, g, p\}$	analytic jitter corresponding ti pdf r

The client measures the average jitter of the arriving packets. We get the ith packet's sending and receiving instants t_i and r_i and we take the mean over all N packets

$$J_m = \frac{1}{N}\sum_{j=1}^{N} |(t_i - t_{i-1}) - (r_i - r_{i-1})|$$
$$= \frac{1}{N}\sum_{j=1}^{N} |R_i - O_i|. \quad (1)$$

This is possible because of the definition of jitter (11) given in Appendix C. In this equation, the values of r_i is known from the local clock and that of t_i from the TCP time stamp. The jitter is simply the difference between the inter-arrival time at the source minus that at the destination so that the synchronization offset between the two ends simply cancels out.

The source, on the other hand,

1. Measures the mean and standard deviation of the inter-arrival process
2. Measures the average transit time
3. Assumes some two-parameter distribution for the inter-arrival time distribution selected from some pre-defined list. For each distribution in the list
   (a) Computes the two parameters of the distribution, generally the location and scale
   (b) Computes the average service time for the queue
   (c) Computes the jitter for the corresponding queue
   (d) Compares the computed jitter with the measured value from the client
4. Chooses the distribution that has the smallest difference between the computed and the measured value of jitter.

We discuss each step and show how it can be done and the amount of work that is needed in each case.

3.1.1 Measuring the arrival parameters

Step 1 is to compute the mean m and standard deviation σ of the packet inter-arrival time using information on the departure times t_i from the server. We have

$$m = \mathrm{E}(R_i) = \mathrm{E}(t_{i+1} - t_i) \tag{2}$$

$$\sigma = \sqrt{\mathrm{var}(R_i)} \tag{3}$$

3.1.2 Measuring the transit delay

In principle, measuring the packet transit delay Step 2 is straightforward: Simply put a time stamp at the source and compare with the arrival time at the destination. This of course assumes that the two clocks are synchronized to a sufficient accuracy. The only practical way of doing this is using the Network Time protocol (NTP) but the accuracy of these measurements can range from a few tens of milliseconds over a network connection up to 100 milliseconds or more on asymmetric routes and in the presence of network congestion. This is not good enough when dealing with network delays of a few tens of milliseconds. For this reason, we use only the RTT where the server uses the `Tsecr` field of the ACK packets which contains the time when the packet being acknowledged left the server. The RTT is then the difference between the time when ACK packet is received back at the server and the departure time of that acquitted packet. From this, one can easily compute the average transit time.

3.1.3 Selecting a candidate inter-arrival distribution

The Step 3 is to choose a two-parameter distribution f_R for the inter-arrival time from some given small set. In this paper, we have used three distributions, log-normal, gamma and Pareto Type I. The corresponding pdf are denoted f_R^l, f_R^g and f_R^p.

3.1.4 Compute the distribution parameters

Given the values of m and σ, the server can compute in Step 3a the scale and location parameters of f_R. In some cases, such as the gamma distribution, this can be done in closed form. If not, a numerical procedure is needed to solve a nonlinear system of two equations in two variables.

3.1.5 Computing the average service rate

With these informations, the server can compute in Step 3b the average service time. For this, we use the fact that the transit time and the service times are related to the Laplace transform of f_R given by (8) in Appendix B. It is generally not possible to do this analytically and we must use a numerical solver for a set of two nonlinear equations in two variables to get the value of μ. At this point, we have all the parameters of the equivalent queue corresponding to the f_R that was chosen.

3.1.6 Computing jitter

Once we have the equivalent queue, we can then use the results of [16] to compute the average jitter $\hat{J}$ for this queue given by (12) in Appendix C. In some cases, we can compute an analytic expression for the jitter. If not, this can be done by a numerical integration technique. We can write $\hat{J}^r$, $r \in \{l, g, p\}$ to indicate that the value of $\hat{J}$ depends on the choice of f_R.

3.1.7 Selecting a model

The server repeats Steps 3a–3d for a number of arrival distribution with corresponding jitter values $\hat{J}^l$, $\hat{J}^g$ and $\hat{J}^p$. Details about the analytical formula are presented in Appendix D. These values are then compared with the actual average jitter value J_m measured by the client and transmitted to the server. The distribution that has the best smallest absolute difference between the computed and measured jitter defines the equivalent queue.

3.2 Summary

The procedure outlined above is such that for a given TCP session in a stationary state and with low loss, the two ends will be able to identify an equivalent queue that 1) has the same first two moments as the arrival distribution, 2) has the same average transit delay and 3) has the the best fit for the jitter measured at the destination among the set of potential arrival processes. The algorithm methodology is summarized in Figure 4.

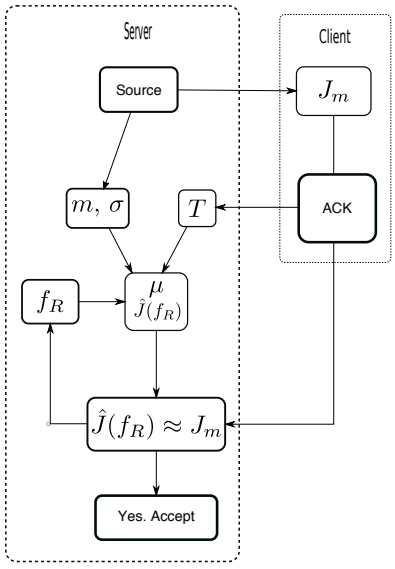

Figure 4: Verification procedure

4 Comparing video streams

The technique we propose here will provide a best guess, in the sense defined above, for the service time and inter-arrival time distributions of the queue. In this section, we compare the equivalent queues obtained for different experimental streams.

4.1 Experimental setup

We have used two different configurations. The simplest is where the server and the client are directly connected to a home router, as shown on Figure 5. While this configuration does not represent in any way a network connection, it is shown here as a benchmark to measure the impact of the access network on the measurements. The other configuration is through the Internet as shown on the top of Figure 1.

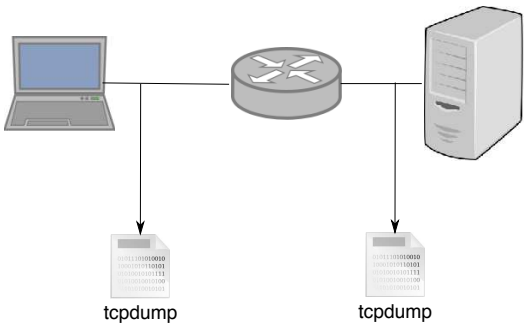

Figure 5: Video transmission test-bed in LAN and WAN

For each experiment, we use about 10 minutes of two MPEG-4 videos which produce traces of 50 000 packets. The streams contain a mixture of scenes, some with more changes than other. These are streamed using the `Videolan` application both as the server and the client and are transported over TCP.

In practice, the algorithm that we described would need the source and destination applications to make the various measurements and calculations required by the model. Because we did not want to modify `Videolan`, we captured the packet traces using `TCPdump` on the wifi or ethernet port of the source and the destination computers. Both traces are analyzed off-line using standard statistical tools to get the values needed by the model.

Except for the first experiment on a single router, the client and server were located in different places. Many were in the Montreal area but we were able to get some measurements from some widely separated locations. The details are given in Table 1.

Table 1: Video streaming over TCP experimental tests summary

Experiment	Client Access Network	Server Location	Client Location
V-1	WiFi	Montreal	Montreal
V-2	WiFi	Waterloo	Montreal
V-3	WiFi	Montreal	Montreal
V-4	$3G^+$	Montreal	Tunis
V-5	WiFi	Milan	Montreal

4.2 Stream measurements

For each stream, we measure four quantities: the mean and the standard deviation of the arrival process, the mean transit time $T = \frac{RTT}{2}$ and the measured jitter J_m. These are summarized in Table 2 and they are given in milliseconds. One can see that for a given video source, the measured values of the mean m are reasonably close to each other but that those for the standard deviation σ show much larger variations

due to the other processes running on the machine. The differences in the transit times and jitter reflect the conditions of the network at the time the measurements were made.

Table 2: Measurements of video stream parameters in milliseconds (msec)

	Video 1				Video 2			
Experiment	m	σ	T	J_m	m	σ	T	J_m
V-1	6.16	43.34	1.985	0.52	13.80	45.00	2.35	4.07
V-2	6.13	38.48	10.02	2.015	—	—	—	—
V-3	6.99	33.35	16.23	2.78	9.18	39.60	15.86	4.15
V-4	7.85	66.15	103.93	12.95	7.29	37.14	86.7	5.07
V-5	6.98	31.90	147.8	9.28	8.29	36.80	83.64	4.69

4.3 Model selection

For each measurement of Table 2, the procedure will yield a best queue based on the difference between the calculated jitter for each distribution $\hat{J}^r$ $r \in \{l, g, p\}$ and the actual measurement J_m. This jitter values are compared in Table 3. The shaded cells of the table show the distribution selected by the algorithm. The interesting point is that in most cases, the log-normal or the gamma distributions are selected. In fact, the log-normal fits many cases and there is a small difference in the other cases.

It will also produce a different mean service time in each distribution case S^r with $r \in \{l, g, p\}$, as can be seen in Table 4. So that the queue utilization ρ^r, $r \in \{l, g, p\}$ is also very different as shown in Table 5.

The results of Table 3 raise an interesting possibility. Instead of trying to guess the distribution for the arrival process, one might choose one distribution for *all* streaming video over TCP. This is a very strong assumption that would need to be validated over a large set of sessions, something that is clearly outside the scope of this paper. Using the results of Table 3, we compute the overall root-mean-square error between the measured and computed values for all distributions. This value is given in Table 6 for the three distributions, where we have excluded the first row since it does not correspond to a real network connection. We can see that the agreement is better for the log-normal distribution than for a gamma or Pareto so that would be a good candidate.

Note however that while the log-normal seems more accurate, using the gamma would be faster. This is because we can express the parameters k and θ analytically in terms of the mean and standard deviation and also because we have a closed form expression (15) for the jitter $\hat{J}^g$ (Appendix D.2).

5 Real-time video over UDP

The case of real-time video is different since these connections often use UDP instead of TCP and the traffic flow is similar in both directions. Instead of having one node identified as the sender and the other as the receiver, as with TCP, in the present case, both end points can be viewed as a sender for its outgoing traffic and a receiver for the incoming packets. In this section, we verify to what extent the G/M/1 model carries over to video over UDP.

Table 3: Experimental END-TO-END jitter and analytical jitter comparison for TCP traffic

	Video 1				Video 2			
Experiment	J_m	$\hat{J}^l$	$\hat{J}^g$	$\hat{J}^p$	J_m	$\hat{J}^l$	$\hat{J}^g$	$\hat{J}^p$
V-1	0.52	0.98	**0.34**	1.71	4.07	**1.70**	1.07	0.01
V-2	2.015	**2.628**	1.38	3.53	—	—	—	—
V-3	2.78	3.80	**2.7**	4.45	4.15	**4.39**	3.18	5.42
V-4	12.95	**9.78**	5.5	6.33	5.07	6.73	7.02	**5.91**
V-5	9.28	7.55	**8.3**	6.21	4.69	7.43	7.94	**6.49**

Table 4: Estimated service time (ms)

	Video 1			Video 2		
Experiment	S^l	S^g	S^p	S^l	S^g	S^p
V-1	0.8624	0.1848	1.8172	1.656	0.759	2.346
V-2	2.2068	0.7969	4.1684	—	—	—
V-3	3.2154	1.6077	5.2425	3.8556	2.0196	6.6096
V-4	4.1605	3.14	7.536	6.0507	4.3011	6.6339
V-5	5.933	4.9907	6.7008	6.2175	4.4766	7.5439

Table 5: Estimated utilization

	Video 1			Video 2		
Experiment	ρ^l	ρ^g	ρ^p	ρ^l	ρ^g	ρ^p
V-1	0.14	0.03	0.295	0.12	0.055	0.17
V-2	0.36	0.13	0.68	—	—	—
V-3	0.46	0.23	0.75	0.42	0.22	0.72
V-4	0.53	0.4	0.96	0.83	0.59	0.91
V-5	0.85	0.715	0.96	0.75	0.54	0.91

Table 6: Root-mean square error

Log-normal	Gamma	Pareto
2.48	4.9	4.6

5.1 Computing the queue parameters

Consider a flow from the one node, called the sender, to the other, called the receiver. The algorithm of Section 3.1 needs some measurements, some of which are not possible with UDP. Step 1 of the algorithm needs an estimate of the mean m and standard deviation σ of the arrival process, which can be measured by the source. Step 2 needs an estimate of the average transit time. For UDP, this information is not available from the packets since they carry no timing information. One possible estimation is to use the first sent-received packet pair exchange to get this estimate [17]. The calculations in Steps 3a–3c can also be carried out as with TCP.

The main problem is the fact that the receiver cannot compute the actual jitter in (1) since the UDP packets contain no timing information or sequence numbers. This means that the algorithm cannot be used for UDP under the assumption we made about the available information.

Still, we think it is worth checking if the technique proposed in Section 3 used for TCP would give good results for UDP it ever were possible to implement it with some future expanded version of UDP or if the required information could be carried on RTCP packets from the RTP protocol.

5.2 Experimental setup

In order to do this, we use the same LAN/WAN expriment configuration as shown in Figure 5. An important real-time video service carried over UDP is video-conferencing so that we chose to use `Skype`, which is one of the most popular video-conferencing platform. Here again, the `Skype` application was installed on two machines. Details about the experiments are presented in Table 7. The test *S-1* was on a residential local

Table 7: Video-conferencing over UDP experimental tests summary

Experiment	Client Access Network	Client 1 Location	Client 2 Location
S-1	WiFi	Montreal	Montreal
S-2	WiFi	Montreal	Montreal
S-3	3G	Tunis	Montreal

network while for *S-2* and *S-3*, the machines were located on distant networks as shown in Table 7. We ran the video-conference for around 10 minutes so that we capture more than 50 000 UDP packets per experiment.

We must assume also that the receiver has a way to identify packets at both ends and the time where they were sent or received. We can do this by using the `tcpdump` trace at the sender and the receiver since these traces contain a time stamp for each packet. We can uniquely identify the packet from the data field so that it is possible for the receiver to compute the actual jitter, something that is not possible based only on the actual UDP packet contents.

5.3 Estimation algorithm

The parameters of the video source are presented for the `Skype` traffic in Table 8. In the present case, the sources are different since they depend on the actual contents of the conversation and the changes in the image. The main difference seems to be the much smaller standard deviation when compared with the video streams. As one could expect, the mean transit time for S-1 is much smaller than for transmission over a real network since the client and the server are on the same LAN. Overall, these parameters are comparable to those for TCP even though the sources are quite different.

Table 8: Source measurements (ms)

Experiment	m	σ	T
S-1	14.36	41.62	4.03
S-2	3.52	13.9	30.7
S-3	5.73	7.66	42.5

The final step is to compare the measured J_m with the analytical values computed from different distributions given by $\hat{J}^l$, $\hat{J}^g$ and $\hat{J}^p$. These are presented in Table 9 where we have indicated the best fit in gray. We can see that here also the best fit is mostly the log-normal or gamma distribution and then with a relatively small difference between the two. We also present in Table 10 the values for the queue utilization. These also cover a wide range from very low to almost saturated.

Table 9: Experimental and analytical jitter (ms)

Experiment	J_m	$\hat{J}^l$	$\hat{J}^g$	$\hat{J}^p$
S-1	3.15	2.29	1.04	2.01
S-2	2.10	2.54	2.37	2.49
S-3	6.16	4.38	4.9	4.05

Table 10: Utilization

Experiment	ρ^l	ρ^g	ρ^p
S-1	0.11	0.28	0.14
S-2	0.58	0.4	0.82
S-3	0.79	0.75	0.87

6 Impact of the arrival process

The results of the previous sections indicate that the G/M/1 queue presents a good model for an end-to-end video connection either for TCP or for UDP protocols. We have seen the that the measured jitter J_m is quite close to most of the values of the analytical jitter $\hat{J}(f_R)$. This seems to suggest that the jitter in a G/M/1 queue depends mostly on the mean m and standard deviation σ of the inter-arrival process R and the shape of the distribution does not matter all that much.

We examine this idea that the jitter does not depend strongly on the particular shape of the inter-arrival distribution by simulation. We measure the jitter in a G/M/1 queue with a different type of distribution such

as Pareto (Pr), log-normal ($LogN$), gamma (Gm) and tri-modal ($TriM$). The tri-modal distribution combines 10% of pareto random variables, 60% log-normal and the other 40% are gamma. For each simulation, we fix the standard deviation σ of R to the values 0.5, 2 and 4. For each case, we plot in Figures 6, 7 and 8 the jitter as a function of the traffic load $\rho = \lambda/\mu$ where the service time $1/\mu$ is chosen as the time unit.

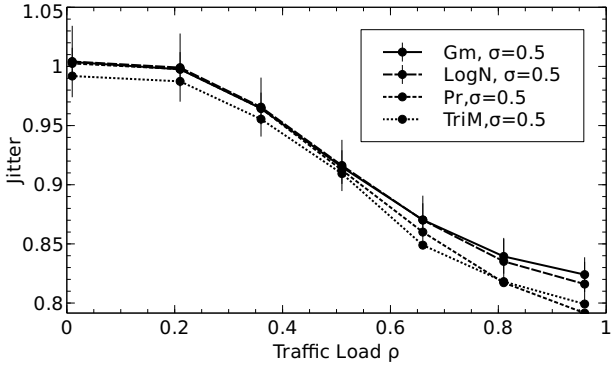

Figure 6: Jitter in a G/M/1 queue for $\sigma = 0.5$

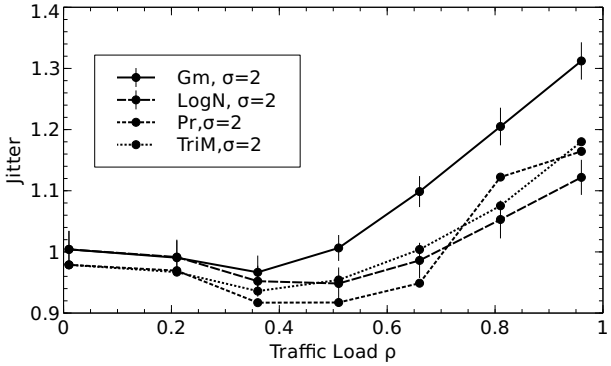

Figure 7: Jitter in a G/M/1 queue for $\sigma = 2.0$

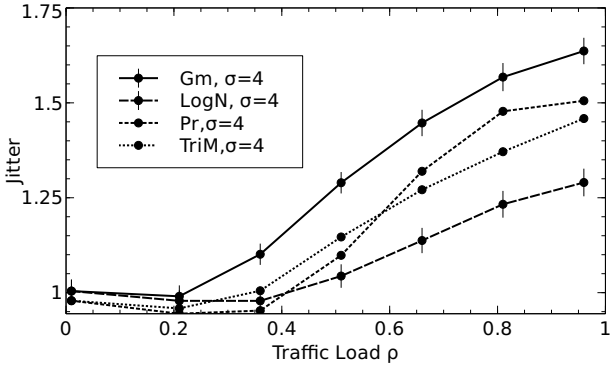

Figure 8: Jitter in a G/M/1 queue for $\sigma = 4.0$

We can see from these results that for a small standard deviation, the shape of the distribution has very little effect on the jitter and this for the whole range of traffic loads.

Results are somewhat different for the two other cases, as can be seen in Figures 7 and 8. The impact of the distribution shape is still small for small traffic load but increases somewhat with the load. Still, we can see that the actual differences are not all that large, even close to saturation.

This result suggests that for the G/M/1 queue, only the inter-arrival mean and standard deviation might be required to identify the queue. This reduces the queue selection process to the estimation of the moments of R for some chosen distribution and the calculation of the jitter for this distribution.

7 Conclusion

A new queue model to study the behavior of IP network video applications was proposed. We found that a single G/M/1 queue can be used as an equivalent queue to study end-to-end network performance of a video session. Moreover, it was found that the log-normal probability density function for packet inter-arrival time gives the best fitting results for end-to-end performance, in particular for jitter. A contribution of this work was precisely to use jitter as a proxy to estimate network delay. This is particularly interesting because, contrary to other latency measurements, jitter measures does not require end-to-end synchronization.

After choosing jitter as a verification tool, we proceeded to compare network jitter in experimental and simulation environments with the results of a G/M/1 queue. We showed that the use of the G/M/1 queue is valid and that, in many cases, the jitter in a queue depends strongly on the mean and standard deviation of the inter-arrival time distribution but not so much on the actual shape of the distribution.

These results are very important and useful for managing QoS for video applications. They yield a good model for over-the-top services that only depend on the video provider and video client, regardless of the details of the network infrastructure. It also gives an analytical platform to determine total service time through the network. Finally, it provides a simple and analytic expression to estimate network jitter for a non-Poisson traffic, which is highly useful for QoS-oriented optimization.

Appendix

A Definition of variables

Based on the above assumption, we now define the variables that we will be using. We define for the i^{th} packet of a particular video stream

λ	arrival rate, $\lambda = 1/E[R]$
S_i	service time,
μ	service rate, $\mu = 1/E[S]$
W_i	waiting time in the queue before service
T_i	transit time through the queue, $T_i = W_i + S_i = r_i - t_i$
η	transit rate $= 1/E[T]$

We denote f_R, f_S and f_T are respectively the probability density functions (PDF) of R, S and T.

B Properties of the G/M/1 queue

In order to characterize the queue, we need a description of the arrival and service processes. The arrival process is defined by the inter-arrival time distribution f_R of packets at the source. We assume that this is known, either because the codec output process is known, or by direct measurement by the source.

The service process is exponential by definition so that the only remaining unknown is the service rate μ that must be estimated from some measurements. For this, we make use of a particular feature of the G/M/1 queue where the service rate μ is directly related to the transit time η through the queue.

First, recall [18] that the transit time T in a G/M/1 queue has an exponential distribution with parameter η given by

$$f_T(x) = \eta e^{-\eta x} \tag{4}$$

$$\eta = \mu(1-\tau) \tag{5}$$

where τ is the probability that a packet will have to wait before entering the server and it is given by the root of

$$\tau = \mathcal{F}_R(\mu - \mu\tau) \tag{6}$$

where

$$\mathcal{F}_R(s) = \int_0^\infty f_R(y) e^{-sy} dy \tag{7}$$

is the Laplace transform of $f_R(y)$.

We see that η and μ are directly related by the following nonlinear system (5) and (6)

$$\begin{cases} \eta = \mu(1-\tau) \\ \tau = \mathcal{F}_R(\mu - \mu\tau). \end{cases} \tag{8}$$

C Jitter definition

The variation of delay in a tagged sequence of packets over time goes under different terms like *jitter*, *end-to-end jitter*, or simply *delay variation*. In the following, we will be using the term *jitter* to mean any of these.

Standards organizations give different formal definitions for jitter. The International Telecommunication Union-Telecommunication Standardization Sector (ITU-T) defines jitter as the variation of delay from some reference value such as the mean or minimum delay [19, 20]. It is the difference between the end-to-end delay, i.e., *latency* of a packet i, T_i, and some reference delay, a_i for a given packet flow

$$J = E\left[|T_i - a_i|\right]. \tag{9}$$

With this definition, it is possible to quickly detect any increase in packet delay.

The Internet Engineering Task Force (IETF) defines the jitter as the difference between two measurement points [21, 22] for a specific packet flow by $G = T_i - T_{i-1}$. The end-to-end jitter is defined as the expected absolute value of random variable G.

$$J = E\left[|T_i - T_{i+1}|\right]. \tag{10}$$

The jitter can be measured without synchronization if we write

$$\begin{aligned} G_i &= T_i - T_{i-1} \\ &= r_i - t_i - (r_{i-1} - t_{i-1}) \\ &= (r_i - r_{i-1}) - (t_i - t_{i-1}) \\ &= O_i - R_i \end{aligned} \tag{11}$$

In other words, the jitter is simply the difference between the inter-arrival time at the destination and at the source for a given packet. We can measure these locally at each end of the connection without having to synchronize them. We can then compute the distribution of G from the difference if we can reliably identify packets in the two ends.

The second requirement is that we should be able to compute the jitter after identifying some queue parameters. Recent work [4, 16] has provided accurate and fast computation models for estimating J derived from Equation 10.

The fact that for a G/M/1 queue both service and transit times are exponential allows us to derive an exact expression for the jitter that depends only on $\mathcal{F}_R$ [16]

$$\hat{J} = \frac{(\eta^2 + \mu^2)}{\eta\mu(\eta + \mu)} + \frac{2}{(\eta + \mu)}\mathcal{F}_R(\eta + \mu) - \frac{1}{\eta}\mathcal{F}_R(\eta). \tag{12}$$

Authors in [16] present simplified expressions for jitter for different inter-arrival time distributions like for Deterministic, Gamma and Pareto distributions.

D Properties of some distributions

Here we review the definition of the pdf and the expression of the mean and standard deviation in terms of the parameters for the distributions we have used for the inter-arrival process. We also give some closed form expressions for the jitter when these are available.

D.1 Log-normal

$$f_R^l(y;\theta,\omega) = \frac{1}{y\sqrt{2\pi}\theta}\ \exp\left(-\frac{(\ln y - \omega)^2}{2\theta^2}\right)$$

$$m = e^{\omega+\theta^2/2}$$
$$\sigma = \sqrt{(e^{\theta^2}-1)e^{2\omega+\theta^2}}$$

In this case, we cannot get an expression for the Laplace transform and we must use numerical integration

$$\mathcal{F}_R^l(s) = \int_0^\infty e^{-sy} f_R^l(y;\theta,\omega)dy \tag{13}$$

The value $\mathcal{F}_R^1(\eta)$ can then be used in (12) to get the jitter.

D.2 Gamma

This is the simplest case from a computational point of view.

$$f_R^g(y;k,\theta) = y^{k-1}\frac{e^{-y/\theta}}{\theta^k\Gamma(k)}$$
$$m = k\theta$$
$$\sigma = \sqrt{k\theta^2}$$

From this, we can get a closed form solution for the parameters k and θ in terms of the mean and standard deviation

$$\theta = \frac{\sigma^2}{m}$$
$$k = \left(\frac{m}{\sigma}\right)^2$$

The Laplace transform can be evaluated analytically

$$\mathcal{F}_R^g(s) = \frac{1}{(1+s\theta)^k}. \tag{14}$$

Next, the simplified analytic expression for jitter is

$$\hat{J}^g = \frac{(\eta^2+\mu^2)}{\eta\mu(\eta+\mu)} + \frac{2}{(\eta+\mu)}\frac{1}{(1+(\eta+\mu)\theta)^k} - \frac{1}{\eta}\frac{1}{(1+\eta\theta)^k}. \tag{15}$$

D.3 Pareto

$$f_R^p(y;\alpha,m) = \begin{cases} \alpha\dfrac{x_m^\alpha}{y^{\alpha+1}} & \text{if } y \geq x_m \\ 0 & \text{otherwise.} \end{cases}$$
$$m = \begin{cases} x_m\dfrac{\alpha}{\alpha-1} & \text{if } \alpha > 1 \\ \infty & \text{if } \alpha \leq 1 \end{cases}$$
$$\sigma = \begin{cases} \frac{x_m}{(\alpha)}\sqrt{\dfrac{\alpha}{(\alpha-2)}} & \text{if } \alpha > 2 \\ \infty & \text{if } \alpha \leq 2 \end{cases}$$

The Laplace transform is given by

$$\mathcal{F}_R^p(s) = \alpha E_{\alpha+1}(sx_m) \tag{16}$$

which we can replace in (12) to get the jitter. Note that, $E_n(x)$ is the exponential integral

$$E_n(x) = \int_1^\infty \frac{e^{-xt}}{t^n}dt. \tag{17}$$

and from this it is possible to compute numerically $\hat{J}^p$.

References

[1] Cisco visual networking index: Forecast and methodology, 2014-2019 white paper, May 2015. [Online]. Available: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/ip-ngn-ip-next-generation-network/white_paper_c11-481360.html

[2] H. Kobayashi and A. Konheim, Queueing models for computer communications system analysis, IEEE Transactions on Communications, 25(1) 2–29, 1977.

[3] E. Rolland, A. Amiri, and R. Barkhi, Queueing delay guarantees in bandwidth packing, Computers and operations research, 26(9) 921–935, 1999.

[4] H. Dahmouni, A. Girard, and B. Sansò, An analytical model for jitter in IP networks, Annals of Telecommunications, vol. 67 81–90, Jan. 2012. [Online]. Available: http://www.springerlink.com/content/5046104335872460/

[5] H. Dahmouni, A. Girard, M. Ouzineb, and B. Sansò, The impact of jitter on traffic flow optimization in communication networks, IEEE Transactions on Network and Service Management, 9(3) 279–292, Sep. 2012.

[6] M. Mandjes, K. van der Waland, K. Rob, and H. Bastiaansen, End-to-end delay models for interactive services on a large-scale IP network, in Proceedings of the 7th workshop on performance modelling and evaluation of ATM & IP networks, 28–30, 1999.

[7] V. Paxson and S. Floyd, Wide area traffic: The failure of Poisson modeling, IEEE/ACM Transactions on Networking, 3(3) 226–244, Jun. 1995.

[8] R. Garcia, X. Paneda, V. Garcia, D. Melendi, and M. Vilas, Statiscal characterization of a real video on demand service: User behaviour and streaming-media workload analysis, Simulation Modelling Practice and Theory, 15(6) 672–689, 2007.

[9] C. L. Ming and J. Schormans, Measurement-based end to end latency performance prediction for SLA verification, Journal of Systems and Software, 74(3) 243–254, 2005.

[10] H. Magri, N. Abghour, and M. Ouzzif, Analytical models for jitter and QoS requirements with IPP and MMPP-2 traffics, in International Conference on Wireless Networks and Mobile Communications. IEEE, 1–6, 2015.

[11] A. Huremovic and M. Hadzialic, Novel approach to analytical jitter modeling, Journal of Communications and Networks, 17(5) 534–540, 2015.

[12] N. Hu and P. Steenkiste, Evaluation and characterization of available bandwidth probing techniques, IEEE Journal on Selected Areas in Communications, 21(6) 879–894, Aug. 2003.

[13] M. Coates and R. Nowak, Network loss inference using unicast end-to-end measurement, in Proc. ITC Conference on IP Traffic, Modeling and Management, 28–1, 2000.

[14] S. Alouf, P. Nain, and D. Towsley, Inferring network characteristics via moment-based estimators, in Proc. INFOCOM, vol. 2, 1045–1054, Apr. 2001.

[15] M. E. Crovella and A. Bestavros, Self-similarity in World Wide Web traffic: Evidence and possible causes, IEEE/ACM transaction on Networking, 5(6) 835–846, Dec. 1997.

[16] H. Dbira, A. Girard, and B. Sansò, Calculation of packet jitter for non-Poisson traffic, Annals of Telecommunication, 71(2) 223–237, May 2016.

[17] D. Rossi, M. Mellia, and M. Meo, Understanding skype signaling, Computer Networks, 53(2) 130–140, 2009.

[18] L. Kleinrock, Queuing Systems. Wiley, 1975.

[19] Internet protocol data communication service—IP packet transfer and availability performance parameters, ITU-T Recommendation Y.1540, Mar. 2011. [Online]. Available: https://www.itu.int/rec/T-REC-Y.1540-201103-I/en

[20] A. Clark, Analysis, measurement and modeling of jitter, Telchemy Incorporated, USA, Tech. Rep., Jan. 2003. [Online]. Available: https://www.telchemy.com/reference/ITUSG12_JitterAnalysis.pdf

[21] C. Demichelis and P. Chimento, RFC 3393: IP packet delay variation metric for IP performance metrics (IPPM), IETF, Nov. 2002.

[22] L. Angrisani, D. Capriglione, L. Ferrigno, and G. Miele, An Internet Protocol packet delay variation estimator for reliable quality assessment of video-streaming services, IEEE Transactions on Instrumentation and Measurement, 62(5) 914–923, 2013.