Simulating Flow Level Bandwidth Sharing with Pareto distributed File Sizes

Julio Rojas-Mora and Tania Jiménez
LIA, University of Avignon,
339, chemin des Meinajaries, Agroparc BP
1228, 84911 Avignon Cedex 9, France.

Eitan Altman
Maestro group, INRIA,
2004 Route des Lucioles, 06902 Sophia
Antipolis Cedex, France.
eitan.altman@inria.fr

ABSTRACT

Our goal is to achieve deeper understanding of the limitation of queueing models, on one hand, and of common simulation practice, on the other hand, as tools for predicting performance of bandwidth sharing between competing TCP flows. In particular, we (i) present an overview of simulation problems that are expected to arise due to the very heavy tail of the distribution of the size of TCP flows, and (ii) through simulations we show that the average sojourn time of competing flows are quite sensitive to various network parameters. The understanding that we get from the first point allows us to better assess when are the conclusions from simulation results on bandwidth sharing reliable.

Using simulations in ns2, we study bandwidth sharing under various load factors and show the benefit of using bootstrap as a post simulation tool for analyzing the simulation results.

Keywords

M/G/1 queue. Processor Sharing. Simulation. Confidence interval.

1. INTRODUCTION

In this paper we present a short overview of difficulties in the evaluation of bandwidth sharing of TCP flows. We first present an overview on the limitations of analytical tools based on flow level queueing models. We then highlight the difficulties in a simulation based approach.

The processor sharing queue has been perhaps the most common model for bandwidth sharing of flows (having the same RTT) of data transfer over the Internet [4], along with the Discriminatory Processor Sharing queue [7] adapted to the case of unequal round trip times. Several research groups have examined its validity through simulations [17, 14, 11, 13]. The conclusions of the papers have not always been encouraging. Indeed,

VALUETOOLS 2011, May 16-20, Paris, France

DOI 10.4108/icst.valuetools.2011.245624

tween the predicted PS model and the simulated model. For example, in [11] the authors write: "We find that the average bandwidth share from our model is close to that obtained from a Processor Sharing (PS) model assuming that the link speed is 75% of the actual link speed."

- In some papers the idea of using a processor sharing type model is criticized. An example is [14] who study processor sharing models in overload. The differences between the simulation and the model's predicted performance goes up to an order of magnitude.
- A more refined model is presented in [23] with an M/G/R PS queue: in that model, the flows do not interact with each other, and the rate at which the packets of each flow are served is not influenced by the number of flows as long as their number is smaller than R. Only when their number exceeds R, the total capacity of the R servers is split equally among the flows. For a small size of flows, it is found that the model underestimates the simulations by a factor of three.

The fact that different contradicting findings have appeared on bandwidth sharing among competing Internet flows can be due either to shortcoming related to simulations and to their interpretation, or to shortcoming in the modeling. As an example for the first possibility, we would cite the authors of [11] who write concerning the heavy tail distribution of the file sizes on the internet: "since a substantial part of the distribution is in the tail, if the simulation is not run for very long the average of the sampled file sizes would be less than the nominal average, thus leading to a lower offered load and hence overestimation of the throughput."

In order to understand the shortcoming of the processor sharing queue as a model for bandwidth sharing we need first to be able to obtain reliable comparisons to simulations or to real measurements. For that, we need to accompany simulations with estimation of the confidence interval. The central limit based confidence intervals may be quite misleading or non-informative as the second moment of the number of flows or of the workload in the system are infinite in stationary regime. We apply therefore an alternative approach to derive confidence intervals and use bootstrap which allows one to obtain more accurate estimation of the confidence intervals and at the same time allows one to accelerate the simulations.

2. BACKGROUND AND RELATED WORK

In the following subsection we present a brief overview of queueing based modeling of bandwidth sharing. They include various papers that identified bad fit when compared to simulations. The simulations themselves to which these models are compared may be unreliable due to the heavy tailed distribution of the flow sizes; this is discussed in the following subsection. We then state the contribution of this paper.

2.1 Modeling bandwidth sharing using processor sharing type queueing models

We have found both good as well as bad fit between the processor sharing prediction for computing the sojourn times and the values obtained by simulations. An extreme example is [13] where an entire order of magnitude of difference is reported between the two. Much better fitness is obtained in some other such as [11], where the differences are of the order of 10%.

2.1.1 Overload conditions

In [14] the authors study a link during a transient overload with TCP connections. (Overload occurs during periods at which the capacity of the bottleneck link is lower than the rate at which new data is generated for connections that use the bottleneck. More precisely, if new flows that use the bottleneck keep arriving at a given rate, say λ per second, and a flow average size (including overheads such as headers) is s, then overload occurs if the bottleneck's capacity is lower than λs.) They consider the case where user behavior doesn't change with congestion, i.e., transfers are neither interrupted nor reattempted. In first place they observe that for different distributions of the size of the transferred file, the number of concurrent connections to the link after congestion starts, grows linearly in time and their slope grows inversely with variance. This behavior is also seen in the rate in which the number of concurrent ongoing flows grows. Comparisons with the PS model show that TCP simulations give much higher growth rates. This is explained by packet retransmissions which affects the effective offered load, and which are not modeled by the PS model.

In PS the increase rate of population overload is known to be sensitive to the distribution of the arrival process only through its mean. The authors show however that there is some effect of the inter-arrival times distribution in TCP concurrent connections growth rate, with worst performances given by distributions with higher variances. The authors say that this behavior can be explained by the slower service rate of transfers arriving in bursts, which stay longer in the system.

It is also observed that for normalized sojourn time, TCP performance degrades at least an order of magnitude faster than PS when files grow in size. This ratio remains constant for all sizes evaluated, thus long connections are as affected as short ones.

For short connections, it is seen for those smaller than 3 packets (packets of 1500 bytes) that sojourn times converge very fast in median to less than 0.5 seconds, but 90% of the connections are below 3 seconds. For slightly larger connections (between 4 and 15 packets), the median is approximately 2.5 seconds while the 90th percentile is more than 15 seconds. This shows that short connections that take long to finish should be restarted as they will finish faster this way.

In the PS model, the normalized transfer time tends to a deterministic limit that grows exponentially in the size of the file transferred. In this paper, simulations show that even after one hour of congestion normalized transfer times are far from converging, with very high variability specially in small connections. This means that small transfers do not take advantage of their size to finish before longer ones as they do in a PS queue. The authors feel that reducing the dispersion of transfer times will improve performance, specially for short connections.

2.1.2 Other models

The M/G/R PS model is used in [23] to study the sojourn time as a function of the file size for one fixed value of RTT. This model underestimates the results of simulations by a quantity that remains quite constant as the size of the file grows. The relative error is seen to be obtained when the size of the file is the smallest. In particular, for the smallest file size considered, the authors find that the model underestimates the simulations by a factor of three.

In [17] the authors consider non symmetric RTTs. They develop a DPS (Discriminatory Processor Sharing) queue model for TPC in the overload regime and test it with simulations in ns2. They obtain an agreement between the expected and the simulated result (with an error of around 10%).

2.2 Simulation involving flows with heavy tailed size distribution

The expected sojourn time of a customer in a processor sharing queue is known to be insensitive to the file size distribution (it depends on the distribution only through the expectation). Its variance, however, is not insensitive anymore, and in particular, it is infinite when the service time of a client (or equivalently in our setting - the size of a file that is transferred) has an infinite second moment [3].

The size of a data transfer over the Internet is known to be heavy tailed [12]. (This is the case for both FTP transfers as well as for the "on" periods of HTTP transfers). Among several candidates for modeling the distribution of these transfers, the Pareto distribution with parameter between 1.05 and 1.5 has been the one that gave the best fit with experiments for the last twenty years, see [8, 2, 6, 15].

This very heavy tail has been causing various serious problems for simulations of data transfers.

An important problem is in the reliability of the results. Standard approach to assess it involve the central limit based confidence intervals. However, this approach is not directly applicable whenever the variance is infinite, which is the case with the stationary sojourn time of a data transfer flow (having a Pareto distribution with a shape parameter K lower than 2). A second problem is the very long duration of simulations needed in order to get good precision.

We present a brief overview of simulation problems encountered in the context of bandwidth sharing among TCP flows

- The authors of [11] write: "since a substantial part of the distribution is in the tail, if the simulation is not run for very long the average of the sampled file sizes would be less than the nominal average, thus leading to a lower offered load and hence overestimation of the throughput."
- Due to the last points one may have problems in inter-

pretation of simulation results . Indeed, various papers in which sharing bandwidth is modeled by processor sharing report differences between the expected theoretical value and the simulation value [13, 14, 11, 17] that vary from around 10% in [11] and go up to a factor of 10 in some situations in [13]. When such deviations occur, it is important to know whether they can be due to the imprecisions in the simulation or to real phenomena.

- The warm-up time is extremely long, see [9].
- One way to avoid heavy tails is to truncate them. This is in spirit of the suggestion in the paper "Difficulties in simulating queues with Pareto service" [10] that says: *"Since for any finite simulation run length, there is always a maximum value of the random variables generated, we, in actuality, are simulating a truncated Pareto service distribution. It has also been argued that there is always a maximum file size or claim amount so, in reality, we are always dealing with truncated distributions."* But where should we truncate the distribution? Truncation at some size M would be valid if the difference between performance with truncation at any other value L that is greater than M has a negligible impact on performance. Simulating with a truncated Pareto distribution may not be sufficient and several other tests of truncation at larger threshold values may be needed. Other approaches are to approximate Pareto with Lognormal or to treat the data from heavy-tailed simulations as transient [24].

2.3 Our contributions

We have presented the background and the work related to the use of processor sharing queues to model bandwidth sharing at the Internet on the flow level. We have put forward the disagreements and the debate related to that model. We also summarized the difficulties in simulating or in interpreting the simulation results due to the heavy tail nature of Internet traffic. The contributions of this paper are first in proposing and testing statistical methods for preprocessing the simulation results that have not been used much in networking. Indeed,

- **Confidence Intervals :** We use the quantile based approach as an alternative to the central limit based approach for obtaining the confidence intervals. The central limit approach is not directly applicable whenever the variance is infinite, which is the case with the stationary sojourn time of a data transfer session (having a Pareto distribution with a shape parameter K lower than 2).
- We then apply this approach directly to the simulation of competing non-persistent TCP transfers of files that have heavy tail distribution. We compare the simulation results to those obtained for the processor sharing queue. We identify various reasons for the deviation of the behavior of TCP from the ideal processor sharing model and manage to quantify the impact of some of these.

Structure of the paper

In Section 3 we present our approach concerning the use of simulation to estimate the average number of packets in the processor sharing queue. We then present in Section 4 simulations of TCP connections that share a common bottleneck link and compare the precision that can be obtained (with and without the bootstrap approach) to the precision obtained in simulating the processor sharing queue. We provide explanations for the differences between the TCP scenario and its corresponding processor sharing model. A concluding section summarizes the paper. An appendix presents some background on Bootstrap and on quantile based confidence intervals.

3. SIMULATING THE QUEUE SIZE OF A PROCESSOR SHARING QUEUE

Through a series of simulations performed in JAVA (available from the authors by request) we exhibit the power of the bootstrap approach: its ability to increase the precision of the simulation of Internet traffic that is throttled by some bottleneck link and at the same time decrease the required duration of the simulation. In this section we restrict ourselves to study of the processor sharing queue which had often been proposed as a model for TCP transfers sharing a bottleneck link in the Internet. A TCP flow is then represented by one customer in the PS queue.

Below we used the PS queue with a Poisson arrival process with a rate of $\lambda = 1$ customers per second. (A customer represents a file when the processor sharing queue is used to model file transfers). We consider three service time distributions: Pareto with shape parameter 1.5, Pareto with shape 2.5, and exponentially distributed. We vary the average service time σ so as to obtain an average load $\rho = \lambda\sigma$ that takes the values 0.6 and 0.7.

Each one of the figures below correspond to the queue size of the processor sharing averaging over 100 independent samples. When considering the PS queue as a model for bandwidth sharing in the Internet, the queue size should be interpreted as the number of ongoing flows.

The duration of each simulation is $6{\cdot}10^6$ sec and there is a warm up time of 300000 sec. In each one of the scenarios described in Figure 1, we present:

1. the theoretical steady state expected queue size
2. the value obtained by the simulations
3. the confidence interval corresponding to a 95% percentile, obtained using the quantile method, and
4. the confidence intervals corresponding to a 95% confidence level after applying the bootstrap method.

We took 100000 resamples out of our 100 original samples for the bootstrap.

Figure 1 displays the precision of the simulations with and without the bootstrap approach for $\rho = 0.6$ (up) and $\rho = 0.7$ (down). The left subfigures are for an exponential distributed service time, the middle and right ones are for a Pareto distributed service time with parameter $K = 2.5$ and $K = 1.5$, respectively. The average service time is the same in the all three sub-figures corresponding to the same ρ. (It was chosen so that indeed $\rho = \lambda \cdot \sigma$ will have the values 0.6 and 0.7, respectively.)

We observe the following points from the simulations:

- The simulations show well the insensitivity of the PS regime to the service time distribution. Indeed, in each

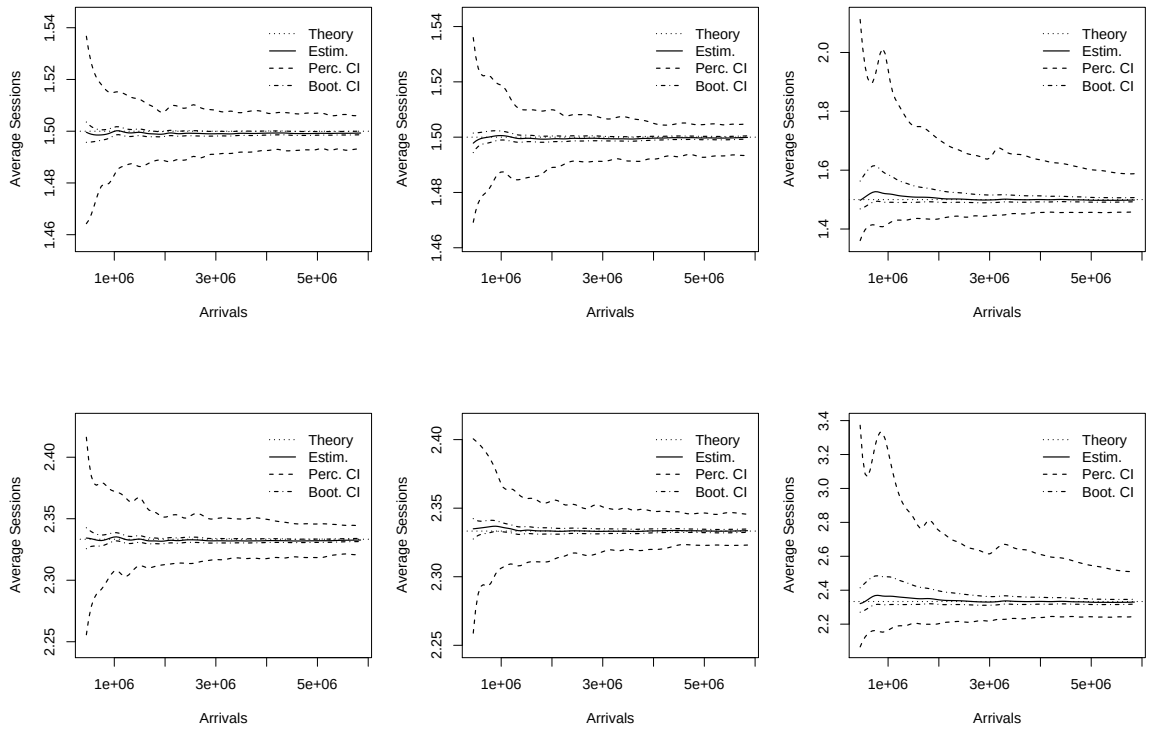

Figure 1: Average number of flows with $\rho = 0.6$ (up) and $\rho = 0.7$ (down). Exponential distribution (left) and Pareto distribution with parameter $K = 2.5$ (middle) and $K = 1.5$ (right)

set of simulations having the same load ρ, we see convergence to the same average rate for the exponential case, the Pareto distribution with parameter $K = 2.5$ and for the Pareto distribution with $K = 1.5$.

- The 95% confidence interval without bootstrap is around 10 times larger than that of the bootstrap (which establishes clearly the advantage of using it). This is seen to hold for any duration of the simulation.

- We see that the confidence interval are around 10 times smaller in the case of exponential and Pareto with shape parameter of 2.5 than in the case of Pareto shape parameter 1.5. This can be expected since the tail of the latter distribution is much heavier.

- Duration of the simulation: For all three distributions, and for the different loads, we see that the precision obtained with bootstrap after already 400000 sec is more than three times better than that without bootstrap after we see that even after 6 million seconds. It thus seems that to get the same precision without bootstrap, one would need to use simulations much longer than 15 times as much as with bootstrap.

4. SIMULATING TCP CONNECTIONS

We compare in this section simulations that we performed with ns2.33 of TCP flows with the simulation of the processor sharing queue. The network we simulate is given in Figure 2.

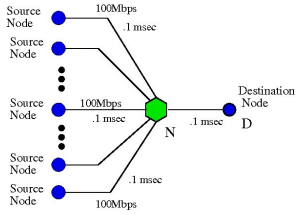

Figure 2: Network Topology

We took 200 input links, each of speed 100 Mbps. The packet size was taken to be 1 KByte. The average flow size was taken to be 200 KBytes. The total round trip delay is 0.4 msec. We simulate the New Reno version without the delayed Ack option. The maximum window size is of 20 packets, which is the default size of ns2. We later change this value.

To make comparisons between precision of simulations that have different event rates and different averages, we find it convenient to normalize the confidence interval. we used the estimated relative half-width (ERHW) of the confidence interval defined as half the difference between the upper and the lower values of the interval divided by the average value.

4.1 Bootstrap and the confidence interval

We are interested in comparing confidence intervals obtained with bootstrap when simulating a processor sharing queue, on one hand, and when simulating TCP, on the other

hand.

Figure 3-5 depict the comparison for $\rho = 0.6$ and 0.7, the first Figure reports simulations for the exponential distribution and the other two are for Pareto distribution with parameter $K = 2.5$ and $K = 1.5$, respectively. Each subfigure contains four curves: The precision (ERHW) obtained by using bootstrap with TCP; The precision (ERHW) obtained in simulating TCP without the bootstrap; The same for simulating the processor sharing queue.

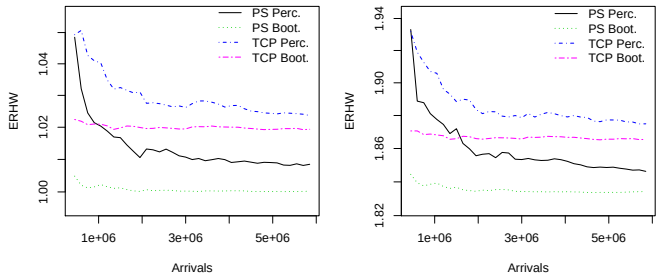

Figure 3: ERHW for exponentially distributed flow size. $\rho = 0.6$ (left) and $\rho = 0.7$ (right).

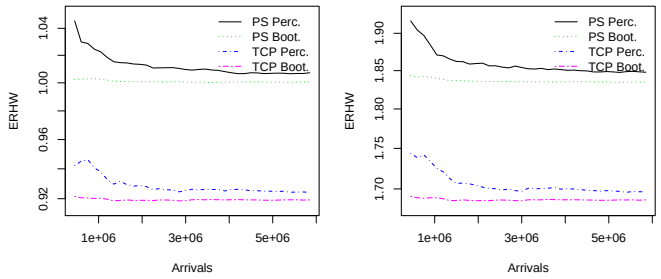

Figure 4: ERHW for Pareto distributed flow size with K=2.5; $\rho = 0.6$ (left) $\rho = 0.7$ (right)

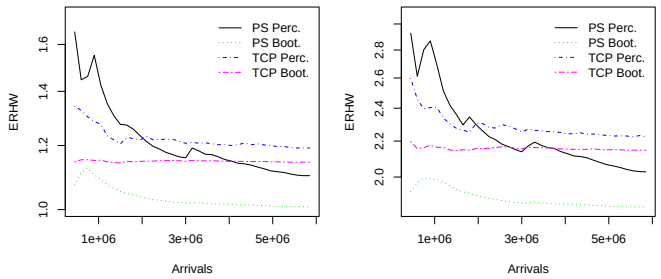

Figure 5: ERHW for Pareto distributed flow size with K=1.5; $\rho = 0.6$ (left) and $\rho = 0.7$ (right)

4.2 Some insight on the bias

As already mentioned, many papers already studied the question of how good a processor sharing queue is able to model bandwidth sharing by TCP connections, see e.g. [13, 14, 11, 17]. They all reported some differences between the expected theoretical value predicted by the processor sharing queue and the simulated value obtained by TCP flows. All reported that for a given load ρ, the actual TCP throughput corresponded to a processor sharing queue with a higher value of ρ. The reported differences vary from around 10% in [11] and go up to a factor of 10 in some situations in [13].

In this section we try to contribute to understanding where the differences come from. We list some of our findings and some recommendations in order to reduce these differences.

4.3 Packet sizes

In ns2, packet sizes are fixed. If we use a probability distribution that has a continuous support (such as the exponential or the Pareto distribution) for the file size, then the actual file size will be a little longer since the last packet will be rounded up. In some applications (see [1]), the average TCP transfer size is around 8-12 KBytes, so if packets of length 1.5 Kbytes are used then the rounding error which is of 0.75 KBytes in the average, will contribute to an increase in ρ of around 7-10 %.

A second source of underestimation of ρ is that in many variants of the ns simulator, when we declare the size of the packet we wish to use then the actual packet size will get 40 bytes added (representing the extra IP and TCP headers). With a packet size of 1 KBytes this will contribute to yet another 4 % underestimation of ρ.

We have incorporated the above considerations in the simulations reported in this paper.

4.4 Burstiness and vacations

Server vacations: It may occur quite frequently that the bottleneck queue is empty but there is at least one ongoing flow. The likelihood of this event increases as the bandwidth delay product increases. Note that for a given ρ, the probability that the system is empty (no flows) is expected to be 1-ρ, so due to the PASTA property, the probability that an arriving flow would find the system empty is also 1-ρ. As long as the arriving flow is the only one in the system, and as long as it is in the slow start phase, the queue at the bottleneck is often empty and the server is then not busy. This can be modeled as a vacation which results in an increase in the workload in the system (as compared to the case in which the server is not on vacation).

Burstiness: We have evidence from simulations that TCP traffic can be quite bursty: many successive packets can belong to the same connection [16]. We suspect that the larger this burstiness is, the less we can use the processor sharing discipline as a model for the flow level evolution; the latter becomes closer to the FIFO discipline. Note that with the Pareto file size distribution with $K = 1.5$, the expected number of customers in a FIFO M/G/1 queue is infinite for every $\rho > 0$. Thus this could explain a larger number of flows.

4.5 Maximum window size

TCP is a window based protocol for reliable communication and congestion control. Each time it has a packet to send, it stamps the packet with a sequence number. To ensure reliability it uses Acknowledgements from the destination to learn about possible losses of packets. The window size indicates the maximum number of packets it can send

before receiving an acknowledgement. The larger the window is, the larger the transmission rate is. In absence of congestion (i.e. as long as Acknowledgements arrive regularly and losses are not detected) the window size keeps growing, until it reaches a maximum size. The default value for this size is 20 packets in ns2. The larger the maximum value is, the more we can expect the connection to be bursty, so we can expect to a larger average number of flows as argued in Section 4.4.

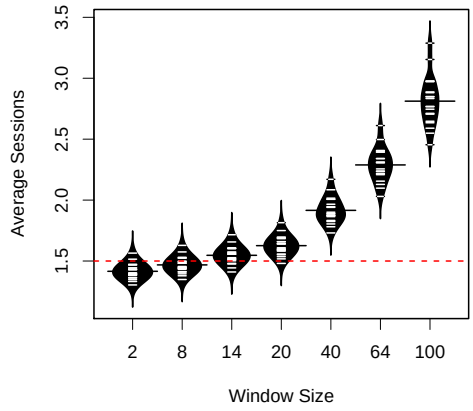

Figure 6: Average number of ongoing flows as a function of the max TCP window size.

We have tested through simulations the impact of the maximum TCP window size on the expected number of ongoing flows and discovered that the latter is indeed sensitive to the maximum window size. The larger the maximum window size is, the larger is the average number of ongoing flows and the average transfer time of a connection. This could perhaps be explained by the burstiness.

We present below our experiments on the impact of the maximum window size on the average number of active flows as well as on other parameters.

Figure 6 reports on the empirical distribution of the average number of ongoing flows as a function of the maximum TCP window size.

For each value of maximum window size, we did 20 simulations. Each simulation lasted till 2000000 arrivals of flows occurred. The 600000 first flows were ignored (this was the warm up time). The other parameters of the simulations are as in Section 4.1.

For each value of maximum window size, we give the empirical probability density of the average number of flows. This is described by the contour of the beanplots (that represents the histogram). Each white bar inside a beanplot represents the average size in one of the twenty simulations. The black bar that traverses each one of the beanplot gives the average obtained from the set of 20 simulations. The dotted horizontal line gives the theoretical average number of customers in the corresponding processor sharing queue.

As we can see, the expected number of ongoing TCP flows that fully agrees with the processor sharing model is the one obtained with a maximum window size of 8. All other values of maximum window size below 20 gave deviations not greater than 10% with respect to the theoretical value. However, we see that for a maximum size of 100, the expected number of flows is almost double the theoretical value.

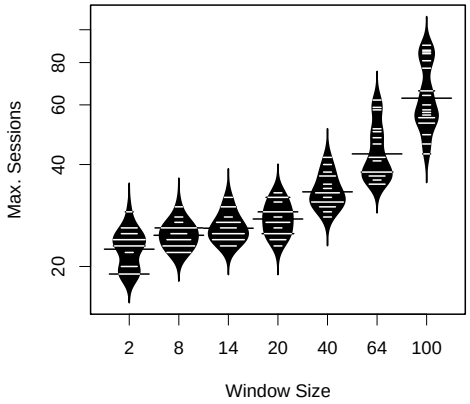

Figure 7: Max number of ongoing flows as a function of the max TCP window size.

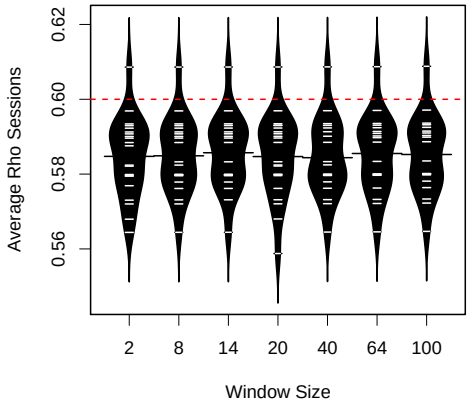

Figure 8: Fraction of arrivals of flows that found the system non-empty upon arrival

Figure 7 reports on the empirical distribution of the maximum number of ongoing flows that were present simultaneously at some time during the simulation, as a function of the maximum TCP window size. Note that unlike the case of average sizes, in which each sample average takes another value, the number of different values of the maximum number of flows that we had within our simulations takes finitely many values, and some values appear several times during the simulations. The number of times that a value appears in the simulations is represented by the length of a white line (and when this value is so large that the line exceeds the boundary of the beanplot, then the bar continuous in black).

Figures 8-9 reports on the empirical distribution of the fraction of arrivals of flows that found the system non-empty upon arrival, and the fraction of arrivals that found the bottleneck queue non-empty. The difference between these in-

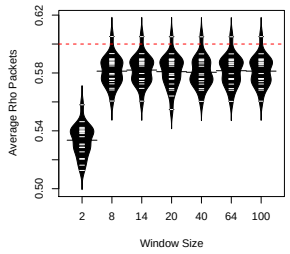

Figure 9: Fraction of arrivals that found the bottleneck queue non-empty, as a function of the maximum window size.

dicate that from time to time there are no transmissions and yet there are ongoing flows. We shall return to that point towards the end of the section.

4.6 Buffer size

The buffer size at the bottleneck queue turned out to be yet another factor that has an influence on the average number of ongoing flows. With a maximum window size of 8 and with $\rho = 0.6$, the size of the buffer for which we obtained full agreement of the average number of flows with the theoretical value (of 1.5) given by the processor sharing was 64. However, we observe that the simulations give good approximations for any larger value of the buffer size, see Figures 10-11. In both figures the largest value of buffer size that we tested was of 1 million packets. We write "INF" in the curves for "Infinite buffer" since with the size of 1 million we had no packet losses, so any buffer of larger size than 1 million would give the same results in this simulation.

The second of these figures uses bootstrap which is seen to result in a considerably better precision.

To understand the deviations from the theoretical value we measure the fraction of time that the queue is empty but there are ongoing flows. We took a maximum window size of 8, $\rho = 0.6$, $K = 1.3$. We obtained around 8% for the case of buffer size of 12 packets, and 0.36% for a buffer of size 64.

We thus attribute the large deviations from the theoretical value to many losses that occur and that result in large periods during which the queue is empty although there are ongoing flows. During these times, the processor sharing queue has "service vacations" and the theoretical results for a queue without vacation are not valid anymore.

This phenomena is countered when using a smaller value of the window sizes and therefore in spite of small buffers one gets better agreement with theoretical results if the maximum window size is smaller.

Note that the "rule of thumb" for selecting buffer size as the bandwidth delay product would give a value of 2 packets in our case which gives values of number of flows much larger than the theoretical value predicted by the processor sharing queue.

For $\rho = 0.7$ we obtained very similar results. The theoretical average number of flows in a processor sharing queue is 2.33. For a maximum window size of 8 we obtained the following values for the average number of flows: 7.41, 3.723 and 2.423 for a buffer of size 12, 24 and 64, respectively.

5. CONCLUSIONS

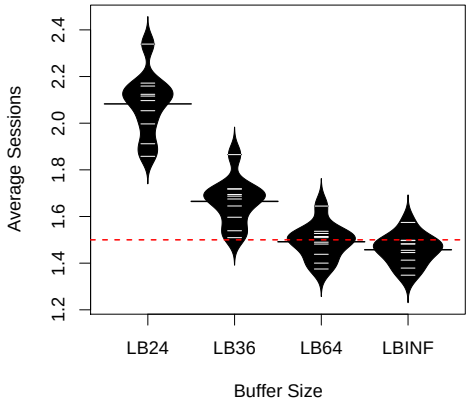

Figure 10: Average number of ongoing flows as a function of the queue size. CI are obtained by the quantile approach.

We have studied in this paper the benefits that the bootstrap method can bring to the simulations of internet traffic sharing a common bottleneck link, and more generally, of the processor sharing queue with heavy tailed service times, which has served as a model for TCP sharing common resources. We found out that due to problems which arise when the central limit theorem cannot be applied, the bootstrap method to calculate confidence intervals is a practical alternative that, at the same time, has the aggregated benefit of substantially shortened simulations. We have analyzed the discrepancy between the results predicted by using the processor sharing queue and those obtained by simulating directly the short lived TCP connections that share a common bottleneck queue. We identified various possible reasons for the discrepancy and provided some recommendations that can help understand and minimize them.

Acknowledgement

The work of the last author was partly suppported by the Semnet project of the joint INRIA Alcatel-Lucent Lab.

6. REFERENCES

[1] K. Thomspson, G. J. Miller, R. Wilder, "Wide area Internet traffic patters and characteristics, *IEEE Network* **6**(11), 1997, pp 10-23.

[2] B. Sikdar, S. Kalyanaraman and K. S. Vastola, "An Integrated Model for the Latency and Steady-State Throughput of TCP Connections", *Performance Evaluation*, v.46, no.2-3, pp.139-154, September 2001.

[3] T. J. Ott, "The Sojourn-Time Distribution in the M/G/1 Queue with Processor Sharing", Journal of Applied Probability, Vol. 21, No. 2 (Jun., 1984), pp. 360-378

[4] J.W. Roberts, "A survey on statistical bandwidth sharing". Computer Networks 45, 319-332 (2004).

[5] K. Tutschku and Ph. Tran-Gia, "Traffic characteristics and performance evaluation of Peer-to-Peer systems". In: Steinmetz, R., Wehrle, K. (eds.) Peer-to-Peer Systems and Applications. LNCS, pp. 383-397. Springer, Heidelberg (2005)

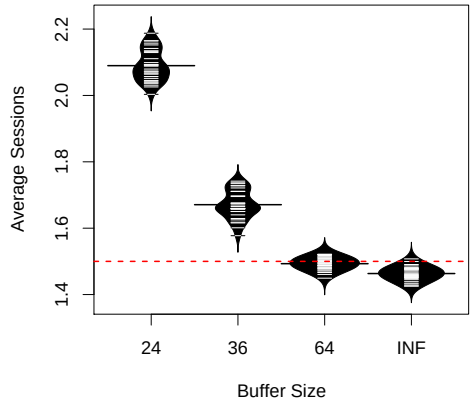

Figure 11: Average number of ongoing flows as a function of the queue size. CI are obtained from the bootstrap.

[6] P. Olivier, "Internet Data Flow characterization and bandwidth sharing modelling", L. Mason, T. Drwiega, and J. Yan (Eds.), ITC 2007, LNCS 4516, pp. 986-997, 2007.

[7] E. Altman, K. Avratchenkov and U. Ayesta, "A survey on discriminatory processor sharing", Queueing Systems, Vol. 53, No. 1-2, pp. 53-63, June 2006.

[8] E. Chlebus and R. Ohri, "Estimating Parameters of the Pareto Distribution by Means of Zipf's Law: Application to Internet Research", IEEE GLOBECOM 2005

[9] Argibay Losada P., A. Suarez Gonzalez, C. Lopez Garcia, R. Rodriguez Rubio, J. Lopez Ardao, D. Teijeiro Ruiz. "On the simulation of queues with Pareto Service". Proc. 17th European Simulation Multiconference, pp. 442-447, 2003.

[10] Gross D., M. Fischer, D. Masi, J. Shorte. "Difficulties in Simulating Queues with Pareto Service." Proc. of the Winter Simulation Conf., pp. 407-415, 2002.

[11] A. A. Kherani and A. Kumar, Performance Analysis of TCP with Nonpersistent Sessions, NCC 2000, New Delhi, January 2000.

[12] A. B. Downey, Lognormal and Pareto distributions in the Internet Computer Communications 28 (2005) 790-801.

[13] N. Dukkipati, M. Kobayashi, R. Zhang-Shen, and N. McKeown, "Processor Sharing Flows in the Internet", 13th International Workshop on Quality of Service (IWQoS), Passau, Germany, June 2005

[14] P. Brown, D. Collange, "Simulation Study of TCP in Overload", Proc. of the Advanced International Conf. on Telecommunications and Intl Conf. on Internet and Web Applications and Services (AICT/ICIW 2006)

[15] S. Floyd and V. Paxson, "Difficulties in Simulating the Internet", Proc. of the 1997 Winter Simulation Conference, Atlanta, GA, 1997.

[16] E. Altman and T. Jiménez, "Simulation analysis of RED with short lived TCP connection", *Computer Networks*, Vol 44 Issue 5, pp. 631-641, April 2004.

[17] E. Altman, T. Jimenez , D. Kofman, "DPS queues with stationary ergodic service times and the performance of TCP in overload", Proc. of IEEE Infocom, Hong-Kong, March 2004.

[18] A. P. A. van Moorsel, L. A. Kant and W. H. Sanders, "Computation of the Asymptotic Bias and Variance for Simulation of Markov Reward Models", *IEEE Proc. of Simulation*, 1996.

[19] P.J. Bieckel and D.A. Freedman. Some asymptotic theory for the bootstrap. *The Annals of Statistics*, 9(6):1196–1217, 1981.

[20] B. Efron. Bootstrap methods: Another look at the jackknife. *The Annals of Statistics*, 7(1):1–26, 1979.

[21] F. Baccelli & P. Bremaud, Elements of Queueing Theory, Springer, 2008.

[22] K. Singh. On the asymptotic accuracy of Efron's bootstrap. *The Annals of Statistics*, 9(6):1187–1195, 1981.

[23] A. Riedl, M. Perske, T. Bauschert, A. Probst, "Investigation of the M/G/R Processor Sharing Model for Dimensioning of IP Access Networks with Elastic Traffic", First Polish-German Teletraffic Symp. PGTS 2000, Dresden, September 2000.

[24] M.C. Weigle. *Improving Confidence in Network Simulations*. Proc. of the Winter Simulation Conference, 2006.

[25] R. Y. Rubinstein and B. Melamed. *Modern simulation and modeling*. Wiley Series in Probability and Statistics. Wiley, New York, 1998.

7. APPENDIX: BOOTSTRAP

Bootstrap is a method created by Efron[20] for non parametrical estimation. Let θ be a parameter of a completely unspecified distribution F, for which we have a sample of i.i.d. observations $X_1, \ldots, X_n$, and let $\hat{\theta}$ be the estimation made of the parameter. From the sample, we will make k resamples with replacement, $X^*_{1,j}, \ldots, X^*_{n,j}$ $\forall j = 1, \ldots, k$, with each element having probability $1/n$ of being selected, and for each resample an estimator $\hat{\theta}^*_j$ will be calculated. By Monte Carlo approximation, the distribution of $\hat{\theta}$ is then estimated by the distribution of $\hat{\theta}^*$. When $k \to \infty$, the estimation of $\hat{\theta}$ will be better and, in turn, the real distribution of θ will also be better estimated. We note that the time and memory overhead for resampling and performing the bootstrap algorithm are often much smaller than the ones needed to create more samples, and can be performed within a very short amount of time. Singh [22], and Bickel and Freedman [19] are good references to understand the asymptotic characteristics of the bootstrap.

Remark. An alternative way to accelerate the simulations is the important sampling or more generally, variance reduction techniques, see e.g. [25, Chap 4] for a general introduction. They are different than the bootstrap approach in that they are based on simulating another model (e.g. use a larger load in order to obtain better estimate of a rare event of reaching a large queue size). Then some knowledge of the system is needed in order to transform the simulated results of the new model to that of the original one. The bootstrap method that we study is a post-simulation approach: it concerns statistical processing of simulated traces. It can be used on top of variance reduction techniques when they

Algorithm 1 Bootstrap algorithm for estimating the mean queue size

1. Make n simulations of the queue size and let X_i, $\forall i = 1, \ldots, n$, be the estimation of the parameter of interest obtained from each simulation.
2. For $j = 1, \ldots, k$ do:
   (a) Let $X^*_{1,j}, \ldots, X^*_{n,j}$ be a resample, with replacement, taken from $X_1, \ldots, X_n$.
   (b) Let $\hat{\theta}^*_j = n^{-1} \sum_{i=1}^{n} X^*_{i,j}$.
3. Calculate $\hat{\theta}^* = k^{-1} \sum_{j=1}^{k} \hat{\theta}^*_j$, the Monte Carlo approximation of the bootstrap estimation of θ.
4. Calculate confidence intervals for $\hat{\theta}^*$ using the quantile method.

are available.

Quantile-based confidence interval

Assume we wish to obtain the confidence interval of the estimation of some parameter of a simulated process X_t. The quantile approach to derive confidence intervals is based on running a number N of i.i.d. simulations (each simulation corresponds in our case to the queue length process or to functions of this process). We then use these to compute the empirical distribution of the function of the random variable.

For $(1 - \alpha) \cdot 100\%$ confidence level, the $(1 - \alpha) \cdot 100\%$ confidence interval by the quantile method is the interval between the $(\alpha/2) \cdot 100$- th and the $(1 - \alpha/2) \cdot 100$-th points of the sorted sample.