Measuring Conditional Web Throughput

Gianluca Mardente, Marco Mellia, Andrea Bianco
Dipartimento di Elettronica - Politecnico di Torino - Italy
e-mail: {mardente, mellia, bianco}@mail.tlc.polito.it

***Abstract*— We focus on the analysis of the performance of HTTP transaction over TCP connections as derived from a measured data set collected at the ingress/egress point of Politecnico di Torino. We argue that looking at global average performance figures may hide some interesting trends that researchers and operators may want to detect. Thus, we partition the original measured set by looking at conditional densities and averages; considered conditional parameters include, among others, TCP connections RTTs, length of data set, adopted loss recovery algorithm. We show that particular care must be taken when analyzing data, since the measured set whose size is reduced by the partitioning scheme could be not large enough to ensure significance. A simple method to evaluate the significance of the presented measures is used. Ingenuity is required to understand behaviors not in line with the classical intuition driven by TCP knowledge.**

I. Introduction

We focus on the analysis of TCP connections in the Internet; we base our study on traffic measurements collected at the ingress/egress point of Politecnico di Torino by using *Tstat*[1], a measurement tool developed at Politecnico di Torino. We are mainly interested in the analysis of TCP throughput. However, when looking at global average performance figures, several specific phenomena may be hidden by the large number of existing TCP connections with largely different characteristics in terms of TCP parameters like RTT (Round Trip Time), congestion control algorithm, length of data transfer, etc. For example, when measuring the average TCP throughput or TCP throughput density, TCP connections using different congestion control algorithm are typically mixed together. Deducing network phenomena or behavior when the characteristic of the involved TCP connections are so different is very difficult, if not impossible. Thus, the classical approach of looking at global averages may hide interesting behavior of particular subsets of TCP connections sharing some common characteristics or a particular set of parameter values. As a consequence, even if the aggregate behavior is important and interesting, it may be worthwhile to partition the original measured set, i.e. looking at conditional density and averages. This may imply re-grouping, after partitioning, TCP connections by "similarity". We aim at answering questions like: which is the throughput difference among TCP connections using SACK instead of Reno loss recovery? Which is the throughput behavior of connections whose RTT ranges from 0 to 10ms, with respect to connections whose RTT ranges from 100ms to 200ms? Which is the range of measured RTT and how many connections show a RTT smaller than 50ms?

This approach may provide several new insights on network performance. Care must be used when analyzing data, since the measured set could be not large enough to ensure significance. We use a simple method to evaluate the significance of the presented measures, so as to eliminate from the analysis measured values which are actually representative of very specific and peculiar situation and not due to general trends.

The most important lessons learned are: i) partitioning the measured set by TCP characteristics is an important and interesting way to analyze measured data; ii) measured data must be analyzed with a lot of care, especially when conditional probabilities are considered; iii) some measured data, even if statistically significant, may show unexpected behaviors, i.e. behavior not in line with the classical intuition driven by TCP knowledge, which require lot of effort to be explained or justified.

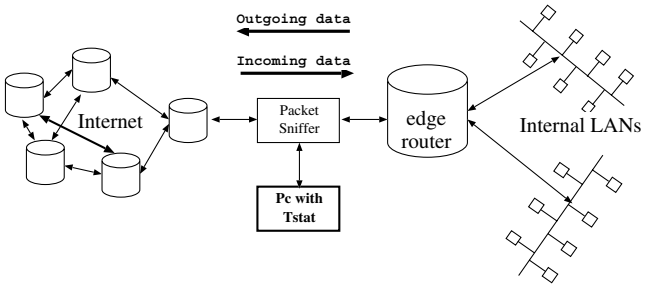

Fig. 1. The measuring setup at the Edge Router of Politecnico di Torino

II. Measuring setup and tool

To collect and organize data presented later in this paper, we use *Tstat*, which is tool that has been developed at Politecnico di Torino since 3 years. *Tstat* is a passive protocol analyzer, that collects measurements at any point in the Internet, with the only obvious constraint of the need of sniffing packets on a link. The detailed description of *Tstat* is available at the *Tstat* home page [1]. Fig. 1 shows the setup we used to collect traces at the ingress link of Politecnico di Torino.

A. Dataset

As sketched in Fig.1, data were collected on the Internet access link of Politecnico di Torino, i.e., between the border router of Politecnico and the access router of GARR/B-TEN, the Italian and European Research network. Within the Politecnico Campus LAN, there are approximately 7,000 hosts; most of them are clients, but several servers are also regularly accessed from the outside. The backbone of our Campus LAN

0-7803-8924-7/05/$20.00 (C) 2005 IEEE

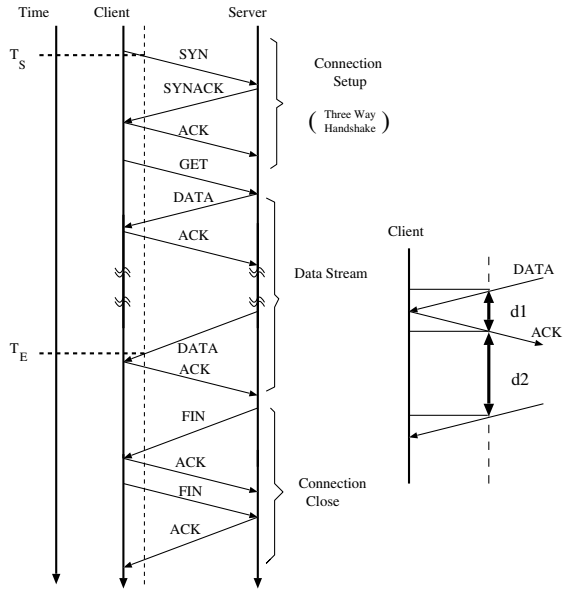

Fig. 2. A sample HTTP TCP connection and the RTT estimation

is based on a switched Fast Ethernet infrastructure. There is a single point of access to the GARR/B-TEN network and, through it, to the public Internet, running at the datalink level an AAL-5 ATM Virtual Circuit.

Among the huge amount of available data, we selected two different time periods:

- JAN.01: from 18/01/2001 to 31/01/2001. The bandwidth of the access link was 16 Mbit/s. About 6 millions TCP connections were tracked in this period.
- OCT.02: from 22/10/2002 to 31/10/2002. The bandwidth of the access link was 28 Mbit/s. About 8 millions TCP connections were tracked in this period.

III. METHODOLOGY

First, note that in the Politecnico Campus LAN the vast majority of TCP connections is originated by internal hosts acting as clients. Thus, even if *Tstat* collects statistics for all connections, in both the description of the measured parameters as well as in the measurements analysis, we will refer to situations in which the TCP client is connected to the Internal LAN, i.e. in the very close proximity of the measurement point. The delay experienced by packets from the measurement point to the internal host can be considered negligible [2], being smaller than 1ms in 90% of cases, since the Internal LAN is always not congested and the number of crossed switches is typically smaller than two. HTTP services account for almost 90% of connections; thus, we concentrate our attention to the HTTP protocol.

Fig.2 sketches the evolution of a generic TCP connection used in an HTTP transaction. After connection set-up, the client performs a GET request, which causes DATA to be transmitted by the server. Note that, if persistent connections are used, several GET-DATA phases (not reported in the figure) may follow. Dashed line represent the time reference at the traffic analyzer which runs *Tstat*.

We list the most important TCP parameters we are interested in, together with the exact definition of the procedure followed to measure them. Some of the parameters are "exactly" measured, with the only limitation that dropped packets cannot obviously be captured; other parameters are estimated on the basis of the information available at the measurement point.

A. Indirect measurements

1) Throughput: The main performance index we are interested in is *throughput*. And, having in mind HTTP service, the "transaction" throughput. We define two time instants:

- T_S: the time at which client request begins, corresponding to the first SYN segment observed from client side[1];
- T_E: the time at which last server data segment over the connection is observed[2].

Referring to Fig.2, we define the average *throughput* of a TCP connection as

$$throughput = \frac{\sum DATA}{T_E - T_S}$$

This definition is fairly intuitive for a single transaction TCP connection as the one depicted in the figure. But it might be wrong considering for example a single TCP connection which carries several transaction, e.g., those caused by persistent connections. This effect will be discussed in the result section.

2) Application latency: Consider the typical behavior of a TCP connection whose sender has a given amount of data to send. According to Nagle's algorithm, data are fragmented by the sender into blocks of Maximum Segment Size (MSS); in case not enough data are available, the sender is allowed to send only one segment smaller than MSS (called tinygram) in every RTT; normally, at the measurement point, we observe a sequence of segments of size equal to the MSS, with the last segment size distributed between 0 and MSS (i.e., all full sized segments, except one tinygram which corresponds to the last segment). However, several TCP connections are detected by *Tstat* in which the sender sends more than one tinygram during connection lifetime. This can be due to server's latency (when the server does not have all data immediately ready for transfer, e.g., HTTP dynamic page); or receiver's latency (when sender sends data relevant to different receiver's request on the same TCP connection, e.g., TCP connections using HTTP persistent connections).

Clearly, the throughput definition provided above is a quite natural choice for the TCP connections of the first type, named "*single flow*". When a TCP connection suffers a latency either at the receiver or at the transmitter, the throughput definition becomes questionable. If the throughput is a measure of network performance, the latency should be taken into

[1]This time is close to the actual time at which the SYN sending event occurs at the client, since the measurement point is close to the client host location.

[2]This time may be significantly different from the time at which the original event occurred at the originating host, but very close to the time at which clients receive the last data segment.

0-7803-8924-7/05/$20.00 (C) 2005 IEEE

consideration and removed from the measurement of the time interval over which throughput is computed. On the other hand, user perceived throughput depends also on the client/server latency. We want to be able to distinguish the two cases, i.e., detect TCP connections that may have encountered a client/server latency during the data transfer and to eliminate or to separately consider those connections.

However, it is not straight-forward to identify whether the data segment size is equal to MSS, as (i) the MSS is not always declared correctly during the three way handshake, (ii) the MSS can change during connection lifetime, and (iii) fragmentation may occur at intermediate router. Thus, we must rely on a heuristic to discriminate "*single flow*" connections from others. *Tstat* therefore evaluates the MSS as the maximum segment size over all connection data segments observed. Given this, we heuristically classify a TCP connection as "*single flow*" when both the following conditions hold

- $\lceil \frac{sent\_data}{MSS} \rceil = sent\_seg$
- $min\_seg = sent\_data - (sent\_segs - 1) \times MSS$

where $sent\_data$ is the amount of data (in bytes) sent by the sender, $sent\_seg$ is the number of data segments sent by the sender and min_seg is the size of the smallest segment sent by sender.

3) Round Trip Time estimation: Fig.2 shows on the right hand side the approach used to estimate *Round Trip Time* (RTT). As already proposed in the literature [3], we compute it as the sum of two components: the first term, $d1$, is the time from the observation of a TCP data segment (from a server) to the observation of the corresponding ACK; the second component, $d2$, is the time from the observation of the previous TCP ACK to the observation of the successive TCP segment sent from the server. We run this estimation of the RTT throughout the TCP connection's lifetime; the connection RTT is then evaluated as the average among RTT measurements during connection life. *Tstat* implements all Karn's rules to avoid problems due to segment loss.

4) Dropping probability: This is a parameter that we must estimate, given that no direct measurement is available. We consider the variable *Out-of-sequence events*, which is the percentage of times a segment whose sequence number is higher than the expected sequence number is received. This measures is related to segment loss probability. Indeed, an out of sequence might occur if (i) a segment reordering appeared in the network or (ii) a segment has been dropped, Given the results in [3], the out of sequence probability is a very good estimation of the dropping probability.

B. Direct measurements

1) TCP flow length: TCP flow length is the amount of data segments transferred by the sender throughout the TCP connection lifetime in the time interval $T_E - T_S$.

2) TCP option: By directly observing the three way handshake, we can measure the percentage of TCP option used during connections. Among the TCP options, we selected the *SACK* [6] capability negotiated between client and server.

C. Measurement confidence

During a measurement experiment, it is very important to know which is the accuracy of results. While there is a common understanding that large datasets allow to get good measurements, we will show, in the next Section, that this is not always true when dealing with Internet measurements. Therefore we use a simple and effective test to assess to accuracy of the measurements. In particular, given a set of measurements $\{x_i\}$ of N samples, which are i.i.d. and with finite variance, the construction of a confidence interval to estimate the significance of the measured parameters involves the estimation of both $\hat{\mu} = \frac{1}{N}\sum_{i=1}^{N} x_i$ and $\hat{\sigma} = \frac{1}{n-1}\sum_{i=1}^{N}(x_i - \hat{\mu})^2$. Let $\mu = E[x]$, and $\sigma = E\left[(x - E[x])^2\right]$. $\hat{\mu}$ is a random variable that, if N is large, is Normally distributed, with average μ and variance σ^2/N,i.e., $\mathcal{N}(\mu, \sigma/\sqrt{N})$. Therefore,

$$Prob\left\{\hat{\mu} - \Delta_{\alpha/2}\frac{\sigma^2}{\sqrt{N}} \le \mu \le \hat{\mu} + \Delta_{\alpha/2}\frac{\sigma^2}{\sqrt{N}}\right\} = 1-\alpha \quad (1)$$

in which $\Delta_{\alpha/2}$ is such that

$$Prob\left\{-\Delta_{\alpha/2} \le z \le +\Delta_{\alpha/2}\right\} = 1-\alpha$$

being z a random variable distributed according to $\mathcal{N}(0,1)$.

Eq.1 shows that if we use $\hat{\mu}$ as estimation of the unknown μ, then the probability that μ is within the interval of extension $\Delta_{\alpha/2}\frac{\sigma^2}{\sqrt{N}}$ around $\hat{\mu}$ is equal to $1-\alpha$. $1-\alpha$ is called *confidence level*, and $\Delta_{\alpha/2}\frac{\sigma^2}{\sqrt{N}}$ is called *confidence interval*.

If the estimation $\hat{\sigma}$ is used instead of the unknown σ, than the $\hat{\mu}$ random variables are distributed according to a $t - student$ with $N - 1$ degree of freedom, and the confidence interval evaluation must be performed accordingly. As the degrees of freedom increases, e.g., larger than 20, the $t - student$ density function approaches the normal density function.

In the sequel, only the measurements for which the confidence interval is not greater than 10% of $\hat{\mu}$ with confidence level of 99% are reported.

IV. Results

We show measurements about both time periods (JAN.01 and OCT.02) when measurements were collected if they are significantly different, while only the most recent dataset will be used in case no major differences can be noticed. Both absolute number of samples (left y-axis) and relative percentage (right y-axis) will be reported, thus giving also the cardinality of the measured dataset, which becomes important when conditional indexes are evaluated.

A. Aggregate measurements

A very "popular" measurement usually considered is the server flow length, i.e., the byte-wise dimension of the payload transported on half connection from server to client. Fig. 3 reports the flow length density histogram in log-log scale and using "number of segments" as unit of measure. A clear linear trend is shown, typical of heavy tailed distributions [4]. While this is no news, we report these measurements to provide an

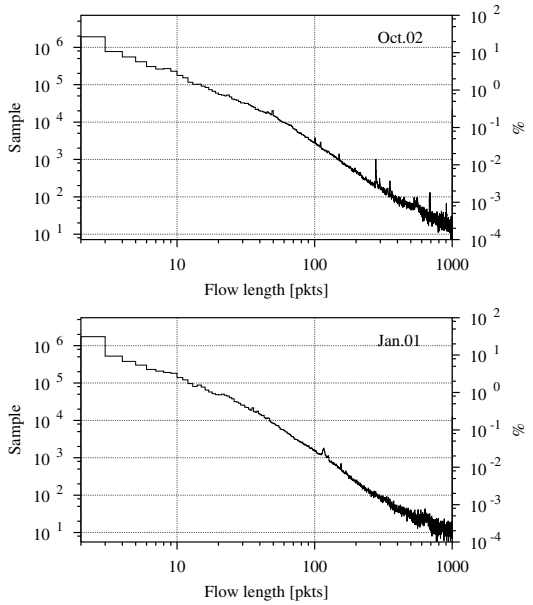

Fig. 3. Server flow length density histogram.

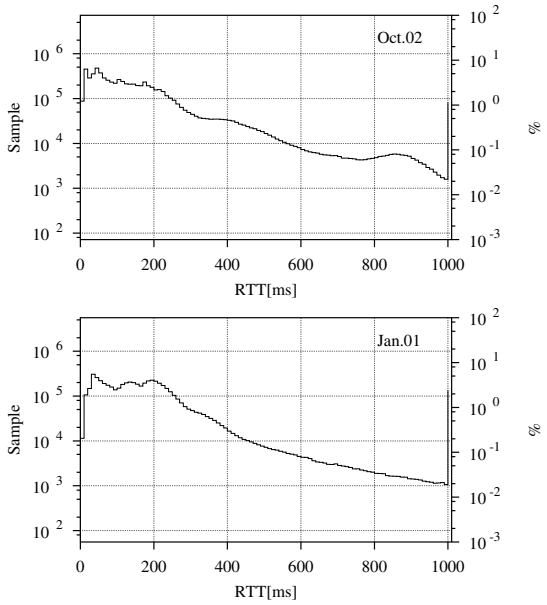

Fig. 4. Estimated average RTT density function.

indication on the amount of available data, as they will be used later when dealing with conditional measurements, giving thus the number of valid measurements.

Recalling that we are showing results about TCP flow statistics which were actively opened by a host inside our campus LAN, the second statistic we report refers to the RTT that might be experienced by a TCP connection, which is reported in Fig. 4. In the JAN.01 dataset, the majority of flows experience RTTs smaller than 500ms. In particular, three major peaks are present, corresponding to RTT smaller than 40ms, in the [100:160]ms range and in the [180:220]ms range. The first peak is due to high speed connections which are terminated at a European server, while the second and third peaks are presumably due to transoceanic servers.

The OCT.02 dataset shows a quite different distribution in the RTT. While it still is possible to identify the peak corresponding to RTT smaller that 40ms, there is only one larger group of connections experiencing RTT in the [60:200] ms range. The pdf tail is also more spread, so that a not negligible subset of connections experience RTT in the [800:900]ms range, which is likely due to paths crossing satellite links.

Fig. 5 reports the transaction throughput. Both plots exhibits a large percentage of connections whose throughput is small, i.e., more than 70% of flows reach throughput smaller than 100kbps. Looking at the density tail, during OCT.02 the experienced transaction throughput is generally higher, with few connections that reach 8Mbps, while in JAN.01 the highest measured rate is 5Mbps.

B. Parameters affecting TCP performance

1) Impact of Selective Acknowledgment (SACK option) and flow length: Fig. 6 plots the first conditional measure: TCP throughput by flow length measured in number of segments. We plot separately TCP flows using SACK option (dashed line), and TCP flows not using SACK (solid line). Note that whereas the percentage of TCP connections using SACK is only about 13% during JAN.01, it increases to 50% in OCT.02. We first observe that the overall throughput increase in the time period OCT.02 is evident also from this graph. Second, SACK provides in general higher throughput, although the improvement is not astonishing. Third, we would have expected to see an increase in TCP throughput as the flow length increases, since short TCP connections should be throttled by the initial transient phase of TCP congestion control; this phenomenon is marginally visible in the measured dataset during both periods. Fourth, several unexpected peaks are visible: the two most evident are in OCT.02 for flow length of 49 segments and in JAN.01 for flow length of 158 segments. This is a very peculiar behavior, since there is no apparent reason why TCP flows composed by 49 segments should obtain a dramatically higher throughput than TCP flows whose data length is 48 or 50 segments. This behavior depends on the fact that the first page of a very popular Italian news server, close to the Politecnico LAN, has a size of exactly 49 segments when using Ethernet dependent MSS=1500Bytes. Given that this conditional measure is dominated by this server, and that the round trip time is very short, this explains the significant higher throughput obtained for flow length of 49 segments. The same reasoning explains the peaks in the measured data set collected during JAN.01. Notice that, even if evidently biased by this, the measurement does fit within the confidence interval!

2) Impact of RTT, flow length and application latency.: Fig.7 plots the transactional throughput of connection that i) have the same amount of segments to be transferred (connections longer than 200 segments are grouped together in the last subset), and ii) suffer the same average RTT (by 10ms

0-7803-8924-7/05/$20.00 (C) 2005 IEEE

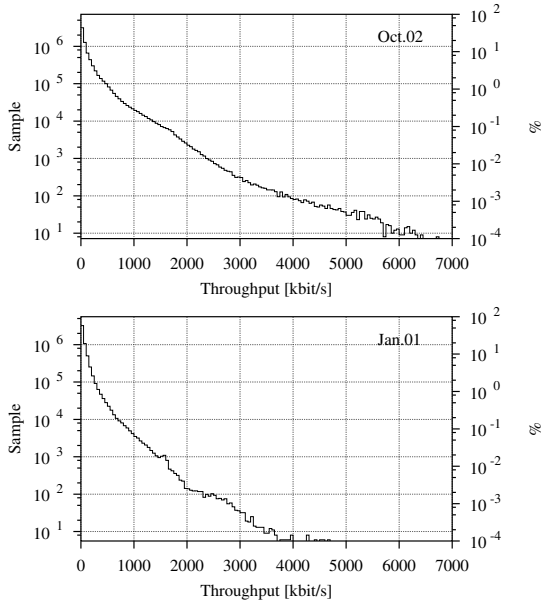

Fig. 5. HTTP transaction throughput density histogram.

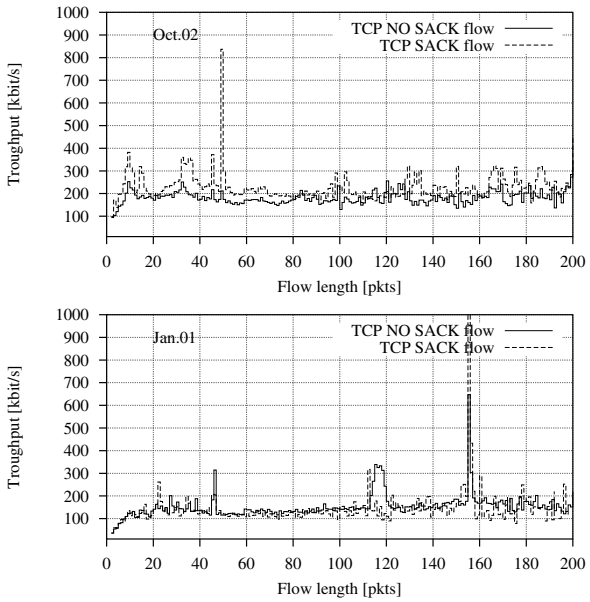

Fig. 6. Average transaction throughput versus flow length conditional to the adoption of SACK option.

intervals). Only OCT.02 dataset is shown. The plot on the bottom considers all connections, while the top plot refers only to "single-flow" connections. First, the confidence test shows that only a limited subset of measurements are accurate, with the "single-flow" condition that restrict the valid samples even more. Notice that the number of samples N in each class is always larger than 100. Nevertheless, the accuracy of the measurements is not sufficient to pass the significance test.

Second, the RTT has a huge impact on the average transactional throughput. On the contrary, the impact of the flow length is not so evident, especially considering connection with RTT larger than 50ms. This is somewhat surprising, given that TCP imposes a congestion control mechanism which slows down the connection especially in the initial Slow-Start phase. More detail on this can be gathered observing bottom plot of Fig.8, which plots, using solid line, the average transaction throughput considering only all connections within the $[10, 20)$ms RTT interval versus flow length. The increase of the throughput with the flow size is visible only up to flows shorter than 10 segments, after which measurements shows almost no dependence versus flow length. Notice again the biased peaks at 49 and other flow length values which were already visible in Fig.5 and 7.

Third, considering only "single-flow" connections (top plot of Fig.7), we notice that the transaction throughput is generally higher than the one experienced considering all connections (bottom plot of Fig.7). This is particularly evident considering connections longer than 200 segments, which are grouped together in the last length class. For these connections, it is again possible to clearly appreciate the impact of the RTT on the average transactional throughput. This suggests that the impact of the application protocol latency has an important role on the user level performance.

To better appreciate this, Fig.8 plots the transactional throughput versus file length for flows that exhibit RTT in the range $[10, 20)$ms. The measurements are also compared with analytical models of the TCP throughput which do not consider the impact of the possible application latency (i.e., "single-flow" connection are considered). In particular, the MAX curve has been obtained considering the transactional throughput a TCP connection of a given length may exhibit when no segment losses are present. It therefore considers only the impact of the window growth mechanisms, and of the initial Slow-Start in particular. It is therefore an upper bound to the actual transactional throughput. The CSA curve instead is a more complex model, derived from [5], in which a finite length always backlogged TCP flow is considered. Taking as input the segment drop probability and the RTT, the model offer an estimation of the transactional throughput. As estimation of the dropping probability, we used the measured out-of-sequence probability, which is shown in Fig.9, that has been proved to be a good approximation of the dropping probability [3], given that the out-of-sequence produced by the network are negligible. Notice that the out-of-sequence probability is almost constant for any flow length. In both models the average measured RTT is used.

Looking at the bottom plot of Fig.8, which considers all connections, a good match between the model and the measurements is obtained for flows shorter than 20 segments. Instead, for flows longer that 20 segments, the CSA model does not provide a good estimation of the transactional throughput, but it always overestimate it. Considering instead only "single-flow" connections (top plot of Fig.8) we notice that a better match between the CSA model and the measurements is present. In this case, the accuracy test filters out a larger

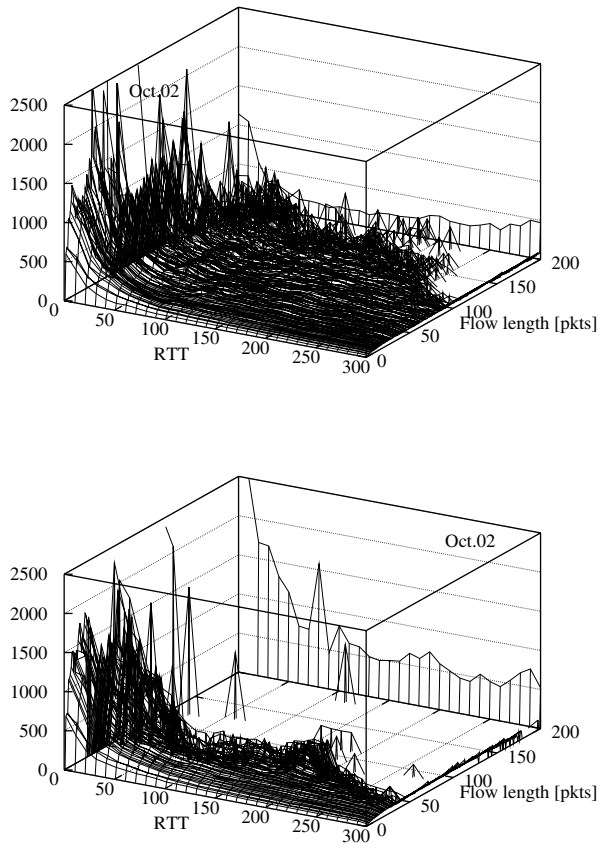

Fig. 7. Average transaction throughput for a given RTT and flow length interval. Considering all flows on the bottom, and considering only flows that do not show any application latency on the top.

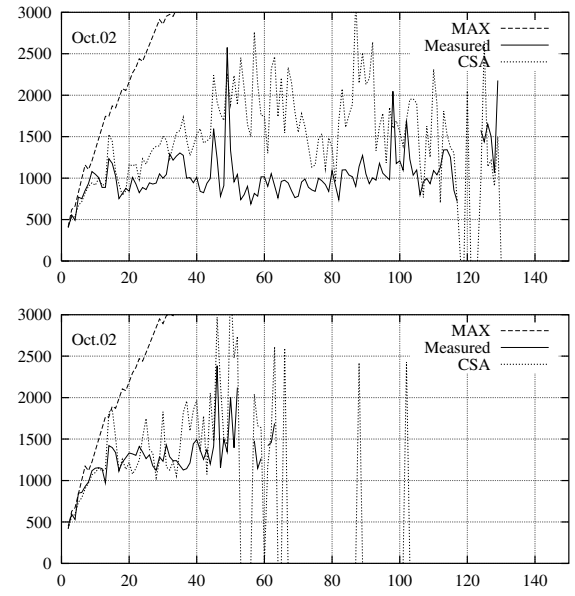

Fig. 8. Average transaction throughput considering flows that have 10ms$\leq$RTT$_i$20ms, versus flow length. Measurement results are compared to the CSA and MAX model. Bottom plot considers all flows, while top plot considers only "single flow" connections. Only OCT.02 data are shown.

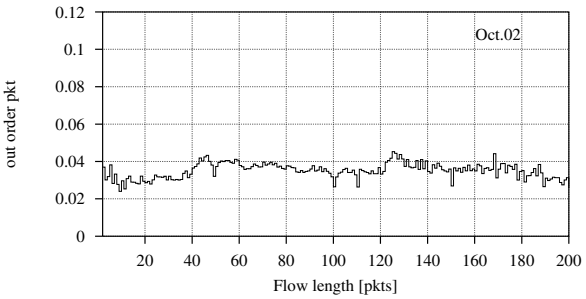

Fig. 9. Average out of order probability versus flow length.

number of samples. This further shows that the application latency plays a big role in the definition of throughput, often throttling down the performance at the user level.

V. CONCLUSIONS

In this paper we focused on the analysis of the performance of TCP connections as derived from a large measured dataset. We inspected the impact of partitioning the dataset into subset having the same properties, focusing in particular on the impact of the flow size, RTT and application latency.

We showed that the transactional throughput is affected by the RTT much more than by the flow size, hinting to the maximum window size today used in the network as possible limiting factor. Moreover, we showed that the latency introduced by the user and application has an important role in the prediction of the throughput user experience. This suggests that the interaction between application level and TCP protocol is an important issue that has been only marginally explored by the research community.

VI. ACKNOWLEDGEMENTS

This work was partly supported by the European Network of Excellence Euro-NGI.

REFERENCES

[1] M. Mellia, A. Carpani, and R. Lo Cigno, "*Tstat* web page," http://tstat.tlc.polito.it/, 2001.

[2] M.Mellia, A.Carpani, R.Lo Cigno, "TStat: TCP STatistic and Analysis Tool", *Quality of Service in Multiservice IP Networks, Second International Workshop, QoSIP2003*, Milano, IT, Feb. 2003

[3] S.. Jaiswal, G. Iannaccone, C. Diot, J. Kurose, and D. Towsley, *Measurement and Classification of Out-of-Sequence Packets in a Tier-1 IP Backbone*. *Infocom 2003*, San Francisco, CA, Apr. 2003.

[4] W. Willinger, V. Paxson, and M. S. Taqqu, "Self-similarity and Heavy Tails: Structural Modeling of Network Traffic," *A Practical Guide to Heavy Tails: Statistical Techniques and Applications, R. J. Adler, R. E. Feldman and M. S. Taqqu, editors*, 1998.

[5] N. Cardwell, S. Savage, T. Anderson, "Modeling TCP Latency", *Infocom 2000*, Tel Aviv, Israel, March 2000.

[6] M. Mathis, J. Madhavi, S. Floyd, and A. Romanow, "RFC 2018: TCP Selective Acknowledgment Options," October 1996.