Analisys and stochastic characterization of TCP flows

J. Aracil, D. Morató and M. Izal
Dept. de Automática y Computación
Universidad Pública de Navarra
Campus Arrosadía s/n
31006 Pamplona, SPAIN
Tel: +34 948 16 97 33
Fax: +34 948 16 92 81
email: javier.aracil@unavarra.es, daniel.morato@unavarra.es, mikel.izal@unavarra.es

September 26, 2000

Abstract

Since the most Internet services use TCP as a transport protocol there is a growing interest in the characterization of TCP flows. However, the flow characteristics depend on a large number of factors, due to the complexity of the TCP. As a result, the TCP characteristics are normally studied by means of simulations or controlled network setups. In this paper we propose a TCP characterization based on a generic model based of stochastic flow with burstiness and throughput (((σ, ρ)-constrains), which is useful in order to characterize flows in ATM and other flow-switched networks. The model is obtained through extensive analysis of a real traffic trace, comprising an approximate number of 1,500 hosts and 1,700,000 TCP connections. The results suggest that TCP connections in the wide area Internet have low throughput while the packet bursts do not suffer an exponential increase, as indicated by the slow-start behavior. On the other hand, the impact of the connection establishment phase is striking. We note that the throughput of the TCP flow is approximately half the throughput which is obtained in the data transfer phase, namely after the connection has been established.

Keywords: Internet services, quality of service, IP over ATM, traffic measurements.

1 Introduction

We are currently witnessing the explosive growth of Internet services such as the WWW. Indeed, recent analysis of Internet traffic traces [1, 2] clearly show that the WWW is the most popular service in the Internet, followed by other services such as the FTP and the email (POP3, SMTP). All of the abovementioned services rely on TCP as the transport protocol and, therefore, it is of primary importance to understand the TCP behavior. Indeed, such protocol has been subject to extensive study in the recent literature [3, 4, 5, 6, 7, 8, 9, 10, 11, 12]. The most part of the articles attempt to characterize TCP behavior. However, the TCP dynamics depend on a large number of issues such as the congestion on the path from the client to server [3], the transmission window negotiated at the connection establishment phase [6], the maximum segment size [7], the roundtrip time estimation and the protocol version (Reno, Tahoe, Vegas) [10, 4], that provide different implementations of delayed ACK and window recovery mechanisms. Thus, due to the TCP daunting complexity we note that the current studies focus on specific aspects of TCP performance in restricted simulation scenarios [3, 4, 5, 6, 7, 8, 9], network links on experimental network configurations [10, 11] and selected client to server links [12].

We note that the analysis presented in the abovementioned studies are constrained by the fact that a real Internet connection is penalized by interfering traffic, which is not captured by simulation models. On the other hand, users navigate the Internet in a random manner. Thus, experimental setups or selected client-server paths do not suffice for the analysis of TCP flows. It is only through unconstrained measurements of real Internet traffic that an accurate characterization of TCP can be achieved.

In [13] a real traffic trace is used to explore TCP dynamics in a real network. However, the analysis is restricted to the relation between ACK compression and segment loss, and their influence to TCP dynamics. On the other hand, trace-driven Internet service analysis is performed in [14] and [15]. The random variables describing per session statistics such as asymmetry, number of bytes and packet interarrival times are analyzed empirically but both studies provide no insight on QoS measurements such as transaction latency and use traces that do not include the most popular service in the Internet: the WWW. Regarding the latter service, most WWW studies are based on logs from servers or proxies [16, 17]. In [18] a trace at client side is collected in order to study WWW session characteristics such as duration, size and user behavior aspects. Since all of the abovementioned WWW analysis rely on connection records obtained with logs produced by the client browser or by the server/proxy we note that only application level statistics can be obtained, and, thus, no emphasis is done on the TCP.

The analysis of real TCP connections is a trade-off between capturing the behavior of TCP flows with a stochastic model or studying a specific aspect of TCP performance in a simulation environment. We note that the characterization of TCP flows in an unrestricted network scenario implies the use of models which are necessarily stochastic, since it is not feasible to model the connection in a deterministic fashion, given the large number of factors that influence the TCP behavior. Ideally, such stochastic model should provide a TCP flows characterization in terms of network parameters, in order to allow for network dimensioning and control. Indeed, there is an increasing interest in IP switching solutions for Internet flows. Tag switching protocols assign a tag to the IP flow at the edge router, so that the routing tables at the backbone routers are simplified significantly, since per-flow and not per-packet routing is performed. The next step in this network evolution towards flow-switching is the allocation of separate resources per flow. In an ATM network such resources could be allocated by the use of Switched Virtual Circuits (SVC). In non-ATM networks it is also feasible to assign separate resources with per-flow queueing combined with scheduling algorithms such as the Distributed Weighted Fair Queueing (d-WFQ). For both cases, we note that a characterization of the TCP flow in terms of the actual network parameters is in order, so that resources can be allocated accordingly.

Precisely, in this paper we present a stochastic characterization of TCP flows obtained from the analysis of a large traffic sample recorded in a real network scenario consisting on an IP-over-ATM link . We derive a network-centric stochastic model for the TCP connection, which is based on a generic traffic flow with burstiness and throughput. At the packet level we consider that the TCP connection can be modeled as a packet arrival process, with X_i bytes arriving in RTT i. While burstiness and throughput provide a TCP characterization amenable for network dimensioning in flow-switched environments the random variable X_i provides information about the dynamic behavior of the flow, as we will see later. Prior to the analysis of the experimental data, let us present the network scenario and measurement tool.

1.1 Network scenario and measurement tool

Our traffic traces are obtained from the network configuration depicted in figure 1. The measurements are performed at the ATM PVC that links Public University of Navarra to the core router of the Spanish academic network (*RedIris* [1]) in Madrid. Rediris topology is a star of Permanent Virtual Circuits (PVCs) which connect the Universities around the country to the central interconnection point in Madrid. From the central RedIris facilities in Madrid a number of international links connect the Spanish ATM academic network to the outside Internet. The measured PVC is terminated at both sides by IP routers. The Peak Cell Rate (PCR) of the circuit is limited to 4 Mbps and the transmission rate in the optical fiber is 155 Mbps.

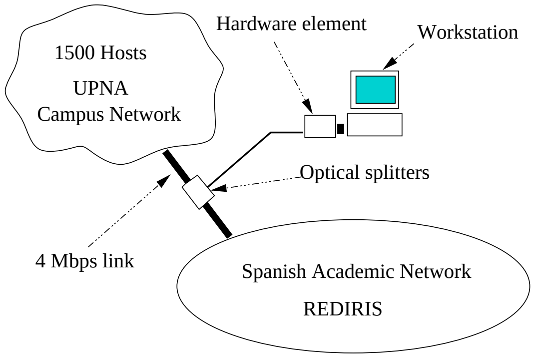

Figure 1: Network measurement scenario

We note that the scenario under analysis is a representative example of a number of very common network configurations. For example, a number of Spanish Internet Service Providers (ISPs) hire ATM PVC links to the operator in order to provide customers with access to the Internet. The same situation arises with corporate and academic networks, that are linked to the Internet through such IP over ATM links. On the other hand, our measurements are not constrained by a predetermined set of destinations but they constitute a real case of a very large sample of users accessing random destinations in the Internet. Furthermore, we carefully check the ATM PVC utilization factor and note that it never reaches 50% during the measurement campaign of one week. This sanity check is performed to ensure that the present analysis accurately portrays a general Internet case. Namely, different connections are facing different bottleneck links according to the destination, but the results are not correlated by a potential bottleneck link in the access. Finally, the wealth of data in the trace provides a strong confidence level in the obtained results. Table 1 summarizes the main characteristics of the traffic trace presented in this paper.

Our measurement tool is implemented with a dedicated hardware to avoid packet filters effects, that can produce measurement skews as noted in [19]. Measurements are performed on-line instead of collecting a trace at the ATM or IP level and performing off-line analysis. The advantage of this

[1] http://www.rediris.es

Start date	Mon 14/12/98 0:00 GMT
End date	Sun 20/12/98 24:00 GMT
TCP conns	1,700,000
IP pkts	9,000,000

Table 1: Trace characteristics

technique is that a large amount of data can be analyzed without interruption since measurements are not so storage-intensive. The dedicated hardware performs extraction of the IP and transport protocol headers, both TCP and UDP. Furthermore, the timestamp resolution for the IP datagram and the ATM cell is 14/12 μs. The offered traffic that can be supported with no cell loss is 300 Mbps. The dedicated hardware relays headers to a UNIX workstation that performs analysis using UNIX threads. One of the major advantages for using threads instead of concurrent processes (namely, processes created with a *fork* system call) is interprocess communication. While the latter may require multiple copies of packet headers in kernel memory the former share the same address space. Therefore, a large throughput between the different threads can be achieved. As a result, a number of threads can be spawned to perform different functions such as TCP connection tracking. Indeed, the measurement tool is able to track the active TCP connections, allowing for a detailed analysis of TCP services, even in high speed environments.

We note that a similar hardware/software approach to perform analysis of large volumes of data is adopted in [1]. However, only connection records are provided by the on-line analyzer presented in [1] (OC3MON) whereas our network monitor records the first 100 packets headers of each TCP connection, thus allowing for a more detailed analysis at the packet level, while keeping the storage requirements at a minimum for long term analysis without interruption. More importantly, a significant advantage of our monitoring tool with respect to OC3MON is that TCP connection teardown is not detected by timeout (64 s.) but the TCP state of every single connection that is captured by the monitor is tracked in real time, according to the TCP segments that are read from the network. Thus, connection establishment and teardown instants are recorded with better accuracy, allowing for a more detailed examination of network parameters such as burstiness and throughput.

2 Stochastic model for TCP connections

In this section we formulate a mathematical model for Internet traffic flows which captures the flow burstiness and throughput [20], that relate to the Maximum Burst Size (MBS) and Sustainable Cell Rate (SCR) as the traffic descriptors at the ATM layer. More formally, the characterization of generic variable bit rate flows has been considered in [20]: Let A_n be the cell arrival process in the time interval n, then the cell arrival process is (σ, ρ)-*constrained* if

$$\sum_{n=k}^{m} A_n \leq \rho(m-k+1)+\sigma \tag{1}$$

for all k, m such that $k < m$. From equation 1 we note that σ is the burstiness and ρ the throughput of the stream A_n. It can be shown that the arrival process A_n will experience zero loss with a single server with buffer capacity σ and service rate ρ [20]. Therefore, the flow characterization given by equation 1 can be used for resource allocation in flow-switched networks. On the other hand, such (σ, ρ) constrains relate directly to the token rate and token buffer size for a leaky bucket traffic shaper. Furthermore, a multiplex of a number of (σ, ρ)-constrained streams is also $(\sigma\prime, \rho\prime)$-constrained, being $\sigma\prime$ and $\rho\prime$ equal to the sum of the single connection's σ and ρ.

A most interesting feature of this model is the throughput ρ is also an indicator of the quality of service perceived by the users for transactional services such as the FTP or the WWW [21]. As a result we provide a TCP flow model that not only provides a flow characterization in terms of network parameters but also serves to the purpose of evaluating the QoS perceived by the user.

Since the connection burstiness and throughput will largely depend on the TCP let us briefly describe the dynamics of such transport protocol. A TCP connection begins with a connection establishment phase in order to establish an initial sequence number and to bind the connection to a unique pair of source and destination ports. On the other hand, the transmission buffer size is negotiated between client and server. Following the connection establishment phase the connection enters slow start. The slow-start algorithm makes transmission from the server be "clocked" by ACKs from the client. Each ACK allows for a one segment increase in the transmission window until the slow start threshold is reached. From the initial exponential increase in window size a linear increase follows (each ACK produces a 1/(window size) increase). Slow start makes TCP transmission behave in a stop-and-go manner until window size reaches a value that allows for continuous transmission, even though packet loss may cause fluctuations of the transmission window. Such value depends on the buffering between client and server and can be approximated by the bandwidth-delay product of the path between client and server, namely the product of the bottleneck link bandwidth and the total roundtrip time (RTT) between client and server. For a throughout description of TCP we refer the reader to [22] and the references therein.

From the above description we note that burst size should increase geometrically in the slow start phase, until the steady-state phase is reached. The connection throughput grows in the same geometric fashion (i. e. 2^i segments per RTT i) until the bottleneck link bandwidth is achieved. Nevertheless, connection throughput depends on the link RTT and packet loss probability [3, 4, 5, 6, 7, 8, 9]. On the other hand, the event that a TCP connection produces a burst of one packet in the first RTT, two packets in the second and so on depends on the RTT, the loss probability and the jitter introduced in the path between client and server, that may separate packets within bursts.

We note that the exact estimation of loss probability and RTT is not feasible. First, loss probability can only be estimated through duplicates of data packets or acknowledgments, but the TCP is not a selective reject protocol and will transmit more packets/ACKs than those actually lost. Secondly, estimates of the RTT based on the time difference between a packet and the corresponding ACK (like the algorithm used by TCP to set the retransmission timer values) provide very poor accuracy since TCP features like delayed ACKs make the server send one ACK per several data packets. Therefore, there is no one-to-one correspondence between a data packet and the corresponding ACK. As a conclusion, we restrict ourselves to a simple estimation of the RTT time scale in order to determine the interpacket gap within a burst.

Such RTT estimation is performed with the initial SYN-SYN handshake. Specifically, we consider the time elapsed from the detection of the SYN from the client to the first segment (ACK to the previous SYN) from the server. We note several advantages of such estimate: first, the server response time has *no contribution at all* in such RTT estimate since the ACK in response to the client initial SYN is sent by the TCP layer. Such server response time, which does not affect the

RTT estimate, is rather significant for *cold* HTTP servers [23], i. e. servers being accessed from the client after an idle period. Even though there is also a *cold-route* effect which does affect our RTT estimate we note from [23] that the cold-server effect is significantly more important. Thus, our RTT estimate provides a RTT-interval value with minimal error. In order to further decrease estimation error a number of probe packets (ICMP echo, for instance) could be sent from the client to the server. However, such ICMP packets should be transmitted either concurrently with the TCP connection from which the RTT is to be estimated or right after connection termination since i) RTT estimation should be performed in a *warm* route for such connection and ii) RTTs are severely affected by time-of-day network activity so they should not be estimated "off-line". To summarize, our RTT estimate, while not completely accurate, provides a simple and accurate RTT-interval value estimate which suffices to slot the time axis in "RTT-cycles". Such normalized time axis serves to analyze and compare the dynamic behavior of a large number of TCP connections with different RTTs.

Secondly, the loss of either the SYN segment from client to server or the ACK from server to client produces a retransmission with a three seconds timer [12]. Surprisingly, we have detected retransmission timers above three seconds, depending on the TCP implementation. This timer is deterministic because there is no way for the TCP protocol agent neither at the client nor at the server side to achieve an RTT estimate at the connection setup phase. The detection of a retransmission event is thus straightforward, so that anomalous RTT estimation can be filtered out easily.

Figure 2 shows the histogram of estimated RTTs. Interestingly, we observe two RTT intervals $(0, 100)$ and $(450, 650)$ milliseconds in which the main part of the samples are obtained. We believe that the first interval corresponds to nationwide and European connections while the second interval corresponds to connections to US servers. Thus, we divide the connection sample into two subsamples according to the estimated RTT intervals, that we denote "low RTT" and "high RTT" in the remainder of this paper.

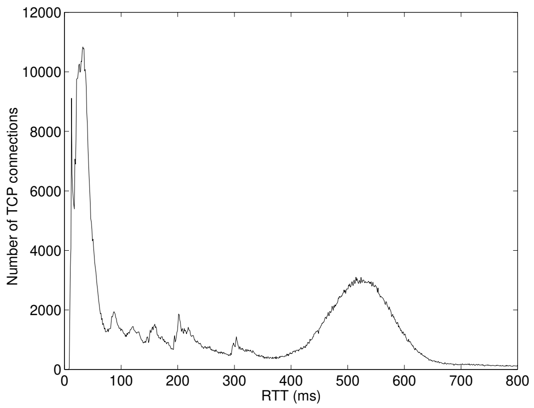

Figure 2: Estimated RTT histogram

Once the RTT estimate has been defined we may proceed to the stochastic characterization of the TCP flow. First we analyze the dynamic behavior of the TCP flow, namely the evolution of the random variable X_i (packets per RTT) with time i. Then we derive the burstiness and throughput parameters according to equation 1.

2.1 Dynamic behavior of TCP connections

The TCP connection begins with slow start and then the steady state phase follows. In order to determine the impact of the initial TCP slow start phase in the total connection duration we first evaluate the number of packets per connection. Figure 3 shows the distribution of number of packets per TCP connection.

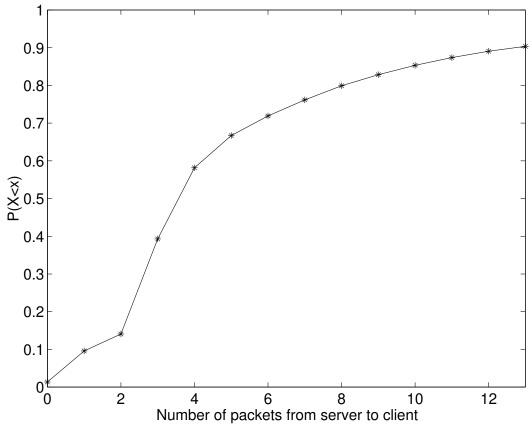

Figure 3: Distribution of number of packets from server to client

We note that TCP connections are mostly short, with 80% of the connections having less than 10 packets. In order to analyze the TCP connection temporal evolution we need to examine how these packets are distributed in the connection lifetime. For example consider a 10 packets TCP connection and recall that the variable X_i represents the packets per RTT i. In the ideal case of no packet loss and large transmission window then X_i takes on the value 1, 2, 4 in the first three RTTs, corresponding to the slow start phase and then the value 3 corresponding to the remaining packets of the connection in RTT 4. Figure 4 shows the average number of packets per RTT from server to client ($E[X_i]$) for connections in the "low RTT" and "high RTT" categories. We observe that the connection is active in packet transmission in the first RTTs, corresponding to the slow start phase and then the packet generation decays with time. On the other hand, we note a significant drop in RTT interval number two, which can be explained with the aid of figure 8. First, we note that 90% of TCP connections in our sample are generated by HTTP clients. For such connections, we note that an ACK from the client to the server is sent in RTT interval number two, followed by the HTTP GET. Then, a file lookup time follows, so the first data packet from the server is likely to arrive at the client in RTT interval number three or four. This rationale is reinforced by the fact that the server may have been accesed by the first time, thus requiring extra time in order to issue a response to the GET PDU [23].

Besides, we note that high-RTT connections show higher variability in the packet generation process due to the larger number of hops from client to server, which makes connection evolution more unpredictable. Overall, the results suggest that the transmission window does not reach the steady-state but rather the most part of the packets are transmitted in the initial RTTs, while the few packets remaining are transmitted in subsequent RTTs.

The results from this subsection show that connections are mostly short and the packet arrival process is dictated by the initial slow start phase. We now turn our attention to the TCP connections throughput and burstiness characterization, that is presented next.

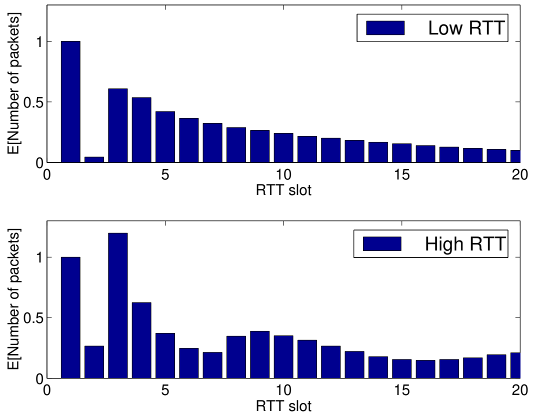

Figure 4: Average number of packets per RTT

2.2 Throughput and burstiness characterization

Connection throughput can be derived simply by the ratio between the transfered bytes and the duration. We distinguish between total duration and duration of the data transfer phase. The former is related to the actual QoS perceived by the user, since it takes into account the time elapsed from the user request to the end of the connection. The latter is the actual connection mean offered traffic to the ATM link. On the other hand, the burst size evaluation requires a threshold for the packet interarrival time in order to determine whether subsequent packets belong to the same burst. Such threshold can be derived easily with the peak rate of the link. Since intra-burst transmission takes place at the peak rate we first consider that several packets belong to the same burst if they are transmitted at the peak rate. Interestingly, we find *no bursts of packets belonging to the same connection* at the peak rate in the measurement period.

We note that bursts within TCP connections depend heavily on the path between the server and the client and the TCP dynamics. First, the network induced jitter may separate packets belonging to the same burst. Secondly and most importantly, since TCP connections are short in size a significant portion of the connection lifetime is devoted to slow start, that produces bursts of geometrically increasing size, namely 2^i packets per RTT round i. A typical TCP connection is expected to produce a burst of one packet, followed by an inactivity interval of roughly one RTT, then a two packets burst, with a packet interarrival time that is determined by the bottleneck link bandwidth and the jitter induced by the network.

This can be clearly observed in figure 5 that shows the histogram of packet interarrival times within connections. We observe low packet interarrival times that correspond to packets within the same burst and packet interarrival times in the vicinity of the RTT value, corresponding to packets in different bursts. Figure 5 provides a threshold for packet interarrival times within a burst. Specifically, We choose a value of 20 milliseconds as the upper bound for the packet interarrival time within the burst. The burstiness derived with this method is always higher than the burstiness derived with the ATM link peak rate, so it provides a safe upper bound for dimensioning purposes.

Figure 6 shows the connection burstiness according to this criteria and figure 7 shows the throughput. The shaded areas correspond to night and weekend measurement intervals, in which the observed traffic is much lower. Figure 7 shows the throughput in hourly intervals and figure 6 shows the probability of i packets per burst also in hourly intervals, where $i = 1, 2, 3$.

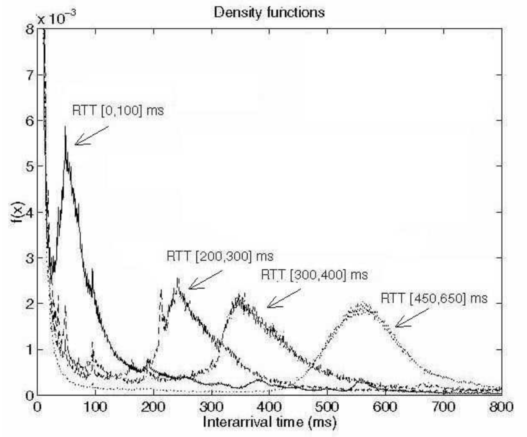

Figure 5: Packet interarrival time (within the same connection)

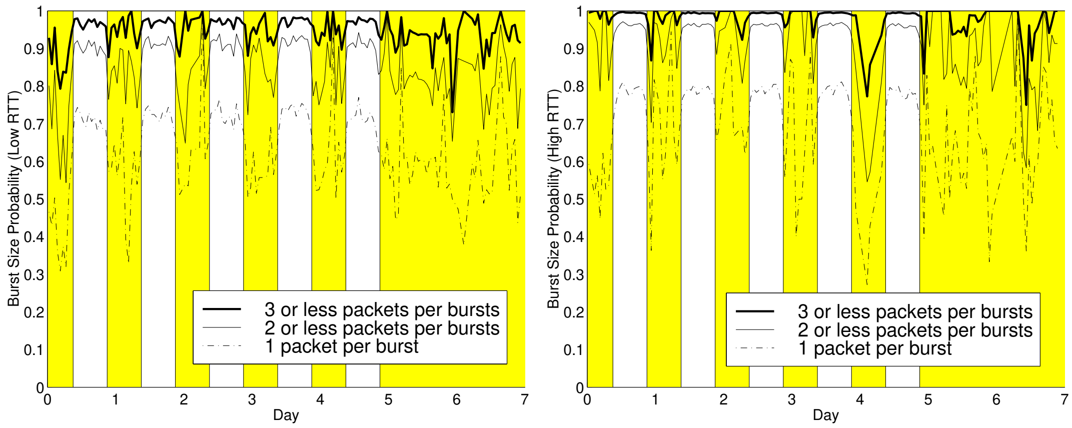

Figure 6: Connection burstiness: $P(burstsize < ipackets)$

Figure 7 shows that the user-perceived throughput, namely the ratio between the bytes transfered and the total duration decreases significantly in comparison to the data transfer throughput. In order to analyze such performance drop, figure 8 shows the connection establishment phase duration in comparison to the data transfer phase duration. The connection establishment time is defined as the time interval from connection initiation (SYN) issued by the client till the connection is in established state in the client side and the first data packet is sent to the server, normally a file request. For the WWW service (90% of our connection sample), such first data segment is the HTTP GET request issued from client to server. Interestingly, the connection setup time does not include the file lookup time in the server, that would presumably take a significant part of connection duration. Even though such lookup time is not included we note two contributions to connection setup delay: First, the URL submission from the client to the server. Secondly, packet loss in connection establishment phase translates into considerable delay since the TCP retransmission timer takes on high values in the connection establishment phase (in the order of seconds) until more RTT samples are received. Furthermore, the impact of the connection establishment phase is significant since connections are short in duration.

On the other hand, the results in figure 6 show a very low connection burstiness (roughly 95%

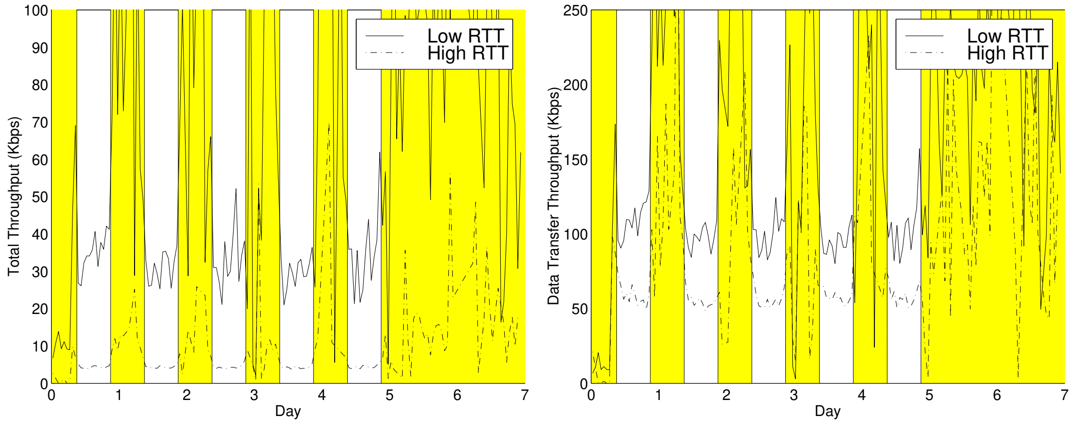

Figure 7: Throughput for total connection (top) and data transfer phase (bottom)

of bursts contain less than 2 packets) and suggests that a small buffer should be enough to provide zero loss in the link. Indeed, we perform on-line simulations of the ATM link under different shaping conditions and verify this hypothesis. We note that the most TCP connections do not reach the steady state phase and, thus, produce short bursts due to slow start, that are also affected by the jitter and loss induced by the network.

3 Conclusions

An empirical study of Internet services on IP over ATM links is conducted in this paper, in order to obtain a stochastic model of TCP flow in terms of network parameters (burstiness and throughput). We note that the link usage is primarily determined by short TCP connections, and so is the QoS perceived by the end user, which can be measured by the connection throughput. The impact of the connection establishment phase in such user-perceived throughput is striking. On the other hand, the TCP slow start dominates the flow lifetime, since only a few packets (less than 10 with 80% probability) are transmitted per connection. Thus, the influence of the TCP dynamics are critical for user-perceived QoS. Besides this throughput characteristics, we note that connections introduce very low burstiness per connection at the network layer.

References

[1] G. Miller K. Thompson and R. Wilder. Wide-area Internet traffic patterns and characteristics. *IEEE Network*, November/December 1997.

[2] D. Morato J. Aracil and M. Izal. Analysis of internet services in IP over ATM networks.

[3] S. Floyd. Connections with multiple congested gateways in packet switched networks, part 1: One-way traffic. *ACM Computer Communications Review*, 21(5), October 1991.

[4] J. Mahdavhi M. Mathis, J. Semke. The macroscopic behavior of the TCP congestion avoidance algorithm. *Computer Communication Review*, 27(3):67–82, July 1997.

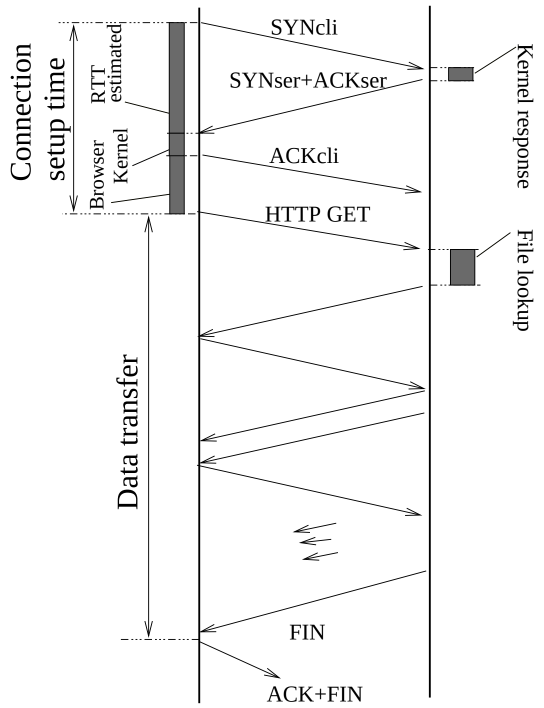

Figure 8: Connection and data transfer phase intervals

[5] T. V. Lakshman and U. Madhov. The performance of TCP/IP for networks with high bandwidth-delay products and random loss. *IEEE/ACM Transactions on Networking*, 5(3):336–351, June 1997.

[6] K. Poduri and K. Nichols. Simulation studies of increased initial TCP window size. RFC 2415, September 1998.

[7] R. Cohen and S. Ramanathan. Tuning TCP for high-performance in hybrid fiber coaxial broadband access networks. *IEEE/ACM Transactions on Networking*, 6(1), February 1998.

[8] K. Obraczka J. Heidemann and J. Touch. Modeling the performance of HTTP over several transport protocols. *IEEE/ACM Transactions on Networking*, 5(5), October 1997.

[9] A. Romanow and S. Floyd. Dynamics of TCP traffic over ATM networks. *IEEE Journal on Selected Areas In Communications*, 13(4):633–641, May 1995.

[10] K. Fall and S. Floyd. Simulation-based comparison of Tahoe, Reno and SACK TCP. *Computer Communication Review*, 26(3):5–22, July 1996.

[11] A. Kumar. Comparative performance analysis of versions of TCP in a local network with a lossy link. *IEEE/ACM Transactions on Networking*, 6(4), August 1998.

[12] S. Savage N. Cardwell and T. Anderson. Modeling the performance of short TCP connections. Available in http://www.cs.washington.edu/homes/cardwell/ quals/quals-paper.ps, October 1998.

[13] J. Mogul. Observing TCP dynamics in real networks. Technical report, Digital, Western Research Laboratory, 1992.

[14] S. Jamin R. Cáceres, P. Danzig and D. Mitzel. Characteristics of wide-area TCP/IP conversations. In *Proceedings of ACM SIGCOMM '91*, 1991.

[15] V. Paxson. Empirically derived analytic models of wide-area TCP connections. *IEEE/ACM Transactions on Networking*, 2(4), August 1994.

[16] S. Glassman. A caching relay for the World Wide Web. In *Proceedings of the First International Conference on the WWW*, 1993.

[17] Marc Abrams, Charles R. Standridge, Ghaleb Abdulla, Stephen Williams, and Edward A. Fox. Caching proxies: Limitations and potentials. In *Proceedings of the Fourth International Conference on the WWW*, 1996.

[18] M. E. Crovella and A. Bestavros. Self-similarity in world wide web traffic: Evidence and possible causes. In *ACM SIGMETRICS Annual Conference on Measurement and Modeling of Computer Systems*, May 1996.

[19] V. Paxson. Automated packet trace analysis of TCP implementations. In *Proceedings of ACM SIGCOMM*, September 1997.

[20] R. Cruz. A calculus of network delay, Part I: Network elements in isolation. *IEEE Transactions on Information Theory*, 37, 1991.

[21] D. Estrin R. Cocchi, S. Shenker and L. Zhang. Pricing in computer networks: Motivation, formulation and example. *IEEE/ACM Transactions on Networking*, 1(6):614–627, December 1993.

[22] W. Stevens. *TCP/IP Illustrated, Volume I.* Addison-Wesley, 1995.

[23] E. Cohen and H. Kaplan. Prefetching the means for document transfer: A new approach for reducing web latency. In *IEEE INFOCOM 00*, Tel Aviv,Israel, 2000.