

Investigation of the M/G/R Processor Sharing Model for Dimensioning of IP Access Networks with Elastic Traffic

Anton Riedl*, Thomas Bauschert*, Maren Perske*, Andreas Probst*

*Institute of Communication Networks, Munich University of Technology (TUM)

*Siemens AG, Munich

Anton.Riedl@ei.tum.de

{Thomas.Bauschert, Maren.Perske, Andreas.Probst}@icn.siemens.de

Abstract

This paper addresses the problem of dimensioning IP access networks for elastic traffic, which is carried by the TCP protocol. The M/G/R Processor Sharing model (M/G/R PS) is used as the basis for capacity computations. The model is extended to take into account TCP specific properties, thus, giving more accurate values, especially for increasing round-trip times. The applicability of the model to TCP Internet traffic is shown through simulations.

Keywords

Network dimensioning, M/G/R Processor Sharing, elastic traffic, TCP

1 Introduction

As the Internet Protocol is becoming the basis of next generation networks, more and more systems are established to provide access to the Internet. Furthermore, competition among Internet Service Providers (ISP) has been increasing drastically, thus, reducing achievable profit margins. While access rates for Internet services are dropping, ISPs still have to maintain a certain degree of service quality to satisfy their customers. As a consequence, exact dimensioning of capacities becomes economically important, particularly for small and medium-size Internet access providers.

Internet traffic can generally be assigned to either one of two fundamentally different traffic categories: stream and elastic traffic. Stream traffic refers to traffic flows whose packets need to be delivered in a timely manner, with packet delay and delay variation being the most important quality measures. Stream traffic is usually generated by (near) real-time applications such as audio/video communications or streaming applications, and carried via the RTP/UDP protocol. Applications that generate elastic traffic require above all reliable packet delivery. Every piece of data needs to be transferred, and in case of packet losses, the respective packets are retransmitted. In terms of quality of service, emphasis is laid on actual throughput. Per-packet delay and delay jitter are rather unimportant, as long

as the total amount of data is delivered within a certain amount of time. The most popular applications of this category are WWW and FTP. They are based on the TCP protocol, which offers reliable data transfer. In addition, TCP employs feedback control mechanisms to adapt the transfer rate to current network conditions, thus, allowing flows on one link to share the available capacity.

In this paper, we address the problem of dimensioning IP access networks for elastic traffic, which is carried by the TCP protocol. We employ the M/G/R Processor Sharing model (M/G/R PS) as the basis for capacity computations and show its applicability to TCP Internet traffic through simulations. Furthermore, we present an extended model that takes into account TCP specific properties and, thus, gives more accurate values, especially for increasing round-trip times.

2 Background and Assumptions

2.1 Relevant TCP Protocol Features

In this section we shortly introduce important TCP protocol mechanisms, which have an impact on the applicability and exactness of the M/G/R Processor Sharing model for TCP traffic. For a complete introduction of TCP, see for example [6].

Connection Establishment

TCP is a connection-oriented protocol where new connections are established following the so-called “three-way handshake”. The connection-requesting instance (usually some sort of client) sends a SYN segment to the server. The server responds to the request by sending its own SYN segment and at the same time acknowledging the SYN of the client. To conclude connection setup, the client has to acknowledge the SYN of the server.

In this paper we do not consider the time spent for connection setup. We only look at the actual data transfer from a (Web) server to the client. At the point where the server starts transmitting data, a connection is already established.

Maximum Segment Size

The Maximum Segment Size (MSS) is the largest amount of payload data that a TCP instance packs into one IP packet. The MSS is negotiated at connection setup, where every end system announces the maximum segment size it expects to receive. Two MSS values, which are often used by default, are 512 and 536 bytes. For Ethernet Local Area Networks the MSS is 1460 bytes.

Nagle Algorithm

The Nagle algorithm is used to reduce the sending of small datagrams. If the amount of data, which is waiting in a TCP send buffer is less than MSS

the data is held back until either more bytes have to be transmitted or all outstanding data has been acknowledged. Only then does the sender transmit the small packet. For bulk data transfer, this means that the last packet of a transmission process might be delayed until the second last packet has been acknowledged.

Sliding Window Mechanism

TCP employs a sliding window mechanism, which limits the amount of data that a TCP instance is allowed to send into the network without having received corresponding acknowledgements. Throughout the paper, we consider the window size to be a multiple of MSS . To allow continuous sending without waiting times, the window size ($w \cdot MSS$) has to be equal to or larger than the bandwidth-delay product, as given in Equation 1.

$$w \cdot MSS \geq C \cdot RTT \qquad \text{Equation 1}$$

C represents the available bandwidth of the TCP connection. RTT is the round-trip time, i.e., the time period between sending out a datagram and receiving the corresponding ACK.

Slow Start Mechanism

The Slow Start Mechanism is a means to avoid network congestion when a new TCP connection is set up and the sender starts transmitting data. Instead of utilizing the maximum possible window size and, thus, injecting a larger amount of data into the network right away, the sender starts out slowly. First, the sender transmits only one segment. For each received acknowledgement it increments the initial window size by one segment. This leads to an exponential increase of the window in case the receiver acknowledges every received packet.

Congestion Avoidance Algorithm

While the slow start mechanism is used to avoid congestion at the beginning of a new connection, the congestion avoidance algorithm applies to packet losses, which may happen during transmission phase. After recognizing congestion in the network, TCP reduces the size of the transmission window, which leads to a decrease of the actual data rate. This establishes the feedback mechanism of TCP.

2.2 Network Model

We consider a hierarchical network architecture as shown in Figure 1. Internet customers are assigned to access areas respective to their location and are connected to the network through dedicated lines. Each access area is served by a Point of Presence (PoP) from where traffic is routed to an aggregation node. At this point, packets coming from several access areas are multiplexed onto one link and transferred to the backbone, either the provider's own or an upstream Internet Services Provider's (ISP) network.

We focus on the star-structured network between the PoPs and the backbone and denote it as the access network.

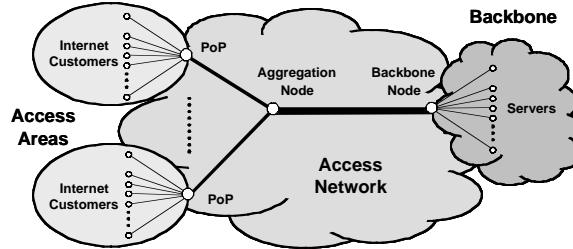


Figure 1 Access Network Structure

2.3 Traffic Assumptions

Our dimensioning procedure is based on IP traffic characterization at TCP connection level without being concerned about the individual packet behavior. It is assumed that every elastic traffic flow is principally related to the transfer of a single file. However, considering the WWW as the main application, one traffic flow might also represent a number of different files downloaded from the same server. The current HTTP protocol specification allows download of several files over the same TCP connection, thus, making the individual transmission processes look like one large one. We assume that new file transfers start according to a Poisson process and that file sizes are heavy-tail distributed, e.g. modeled by a Pareto distribution. The presented formulas can be applied to any file size distribution as they are not sensitive to the shape of the general service time distribution [7]. Furthermore, for simplification, we assume that all Internet users experience approximately the same maximum access bit rate (due to similar modem connections to the access node).

3 Dimensioning Procedure using the M/G/R Processor Sharing Model

3.1 M/G/R PS Model

The idea behind the M/G/R Processor Sharing model (see for example [1]-[5],[7]) in our context is to look at file transmissions over one or more links. A network link is viewed as a Processor Sharing system, which offers its service to several customers by sharing the system performance (i.e., link capacity). All active jobs are served quasi-simultaneously by assigning little time slots to each of them and processing them sequentially. Spoken in terms of file transmission, this is interpreted as simultaneously transmitting files

over one link by breaking them into little pieces (i.e., packets) and transferring these pieces sequentially. The control mechanism of TCP regulates the amount of traffic in a way that resources are shared equally among active flows. Depending on whether one TCP connection is able to utilize the total link capacity by its own, the system is described either by an M/G/1 PS or M/G/R PS model.

In case the attainable transfer rate of one connection is limited to a certain peak rate r_{peak} (e.g., modem bit rate), a shared link appears like a system with R servers where $R=C/r_{peak}$, i.e. the ratio between the link rate C and the peak source rate r_{peak} . Up to R flows can be served at the same time without rate reduction imposed by the system. If r_{peak} is larger than the link capacity the M/G/R PS model reduces to the simple M/G/1 PS model. A nice property of M/G/R PS queues is the independence of the average sojourn time (average time in system) from the shape of the general service time or file size distribution. Large files do not delay small ones too much.

For the M/G/R PS model, the expected sojourn time (or transfer time) $E\{T(x)\}$ for a file of size x is given by:

$$E\{T(x)\} = \frac{x}{r_{peak}} \left(1 + \frac{E_2(R, R\rho)}{R(1-\rho)} \right) = \frac{x}{r_{peak}} \cdot f_R \quad \text{Equation 2}$$

Here, ρ denotes the utilization on the link ($\rho = \lambda_e \cdot x_{mean} / C$ with flow arrival rate λ_e and average file size x_{mean}). E_2 represents Erlang's second formula (Erlang C formula) with $A = R\rho$ as given in Equation 3.

$$E_2(R, A) = \frac{\frac{A^R}{R!} \cdot \frac{R}{R-A}}{\sum_{i=0}^{R-1} \frac{A^i}{i!} + \frac{A^R}{R!} \cdot \frac{R}{R-A}} \quad \text{Equation 3}$$

In Equation 2 the expression in parentheses is also called "delay factor" f_R as it denotes the increase of the average file transaction time imposed by the system. The delay factor is a quantitative measure of how link congestion affects transaction times, taking into account the economy of scale effect.

3.2 Extended M/G/R PS Model

The M/G/R PS model does not consider any queueing or propagation delay within the network and assumes ideal capacity sharing among active flows. However, in practice TCP flows are not necessarily able to utilize their fair share of available bandwidth. TCP's effectiveness of capacity sharing is determined by the TCP algorithms introduced earlier, which are

affected by network conditions like round-trip times and packet loss probability. Especially the round-trip delay influences the degree of divergence between the pure M/G/R PS model and practical achievements. Therefore, we suggest extending the pure M/G/R PS model by some extra terms. The extended model considers the slow-start and, optionally, the Nagle mechanisms, which both do not really comply with the idea of pure processor sharing.

Slow Start Extension

The slow-start mechanism increases the sojourn time by not utilizing the available bandwidth at the beginning of each transmission process. Let us make following assumptions:

- a data load of size x (including overhead) needs to be transmitted
- the maximum window size of the TCP connections is large enough to cover the bandwidth-delay product of the network
- no packet losses happen during slow-start phase.

From Equation 1 we can derive the minimum window size w^* (in multiples of MSS , which in this chapter refers to the maximum allowed packet size including header, often referred to as MTU) so the available bandwidth can be utilized. Given the required delay factor f_R and the bottleneck bandwidth r_{peak} , w^* is given by Equation 4.

$$w^* = \left\lceil \frac{C_{available} \cdot RTT}{MSS} \right\rceil = \left\lceil \frac{r_{peak} \cdot RTT}{f_R \cdot MSS} \right\rceil \quad \text{Equation 4}$$

Assuming a TCP sender starts transmitting with a window size of 1, the size of the window increases exponentially with approximately 2^i with $i = 0, 1, 2, \dots$ (time unit equals RTT). Thus, at time step $n^* = \lceil \log_2 w^* \rceil$ the window is for the first time equal to or larger than w^* .

The amount of data sent up to the time when the sender window reaches or exceeds w^* , i.e., when the sender can start utilizing its bandwidth share, is

$$x_{slow-start} = (2^{n^*} - 1) \cdot MSS \quad \text{Equation 5}$$

If the total size x of the file is smaller than $x_{slow-start}$, the sender never reaches the state of fair capacity sharing. Instead, all packets will be sent out during slow-start phase. The computation of the expected time to send all data is illustrated in Figure 2. The first part gives the sum of the necessary round-trip times, which are mostly spent waiting for acknowledgements as a consequence of the slow start mechanism. The second term considers the time of sending the rest of the data with the available capacity. If all of these

packets arrive at the receiver without errors, the actual data transmission process is finished.

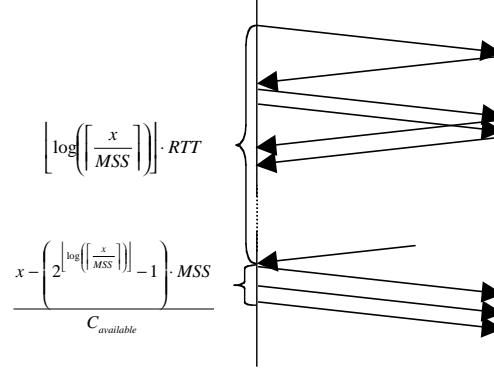


Figure 2 Data Transfer in Slow-start phase

Thus, the expected transmission times $E\{T(x)\}$ for files of size x (including overhead) can be approximated by Equation 6.

$$E\{T(x)\} = \begin{cases} \log_2\left(\left\lceil\frac{x}{MSS}\right\rceil\right) \cdot RTT + E_{M/G/R}\{x - x_{start}\} & x < x_{slow-start} \\ n * RTT + E_{M/G/R}\{T(x - x_{slow-start})\} & x \geq x_{slow-start} \end{cases}$$

Equation 6

where $E_{M/G/R}\{T(x)\}$ is the sojourn time corresponding to the pure M/G/R Processor Sharing model and x_{start} is given by

$$x_{start} = \left(2^{\left\lceil\log_2\left(\left\lceil\frac{x}{MSS}\right\rceil\right)\right\rceil} - 1\right) \cdot MSS \quad \text{Equation 7}$$

Nagle Extension

The model can easily be extended to consider the Nagle algorithm. Since this algorithm just delays the sending of the last packet, one times the round-trip time RTT has to be added to the sojourn time.

3.3 Dimensioning Procedure

Our dimensioning procedure considers the average file transfer time $T_{Tolerance}$ for files of a certain size $x_{threshold}$ as the relevant QoS criterion. This time, which cannot be smaller than the minimum possible transfer time $x_{threshold}/r_{peaks}$, should be restricted by some upper bound respective to a typical Internet user's assumed tolerance. We argue that a user does not care whether

documents of smaller size are transmitted proportionally faster or not, as long as the tolerance level is not exceeded. Therefore, an appropriate file size $x_{threshold}$ (e.g., the 95th percentile of an assumed file size distribution) should be chosen for the dimensioning process. As we will see later, the pure M/G/R PS model works well for larger files. Therefore, Equation 2 can be used to find the required capacity when $x_{threshold}$ and $T(x_{threshold}) = T_{Tolerance}$ are given. For small $x_{threshold}$ or larger RTT , Equation 6 has to be solved.

For an end-to-end connection, which runs over multiple network links only the bottleneck link (i.e., the link with the highest delay factor f_R) determines the transaction bit rate (r_{peak} / f_R) and, thus, the file transfer time. Consequently, optimum network dimensioning means assigning link capacities in a way that each link suffers the same delay factor (which in turn is driven by the end-to-end QoS requirement). As shown in our simulations, the end-to-end QoS criterion is sufficiently met if each link on a path of our access network structure is dimensioned accordingly.

4 Dimensioning Example and Simulation Results

In our simulation scenarios, a total number of 1000 TCP sink nodes represent the customers and are assigned to 10 access nodes evenly. The ten access nodes are connected to the aggregation node from where a link is set up to the backbone node. The TCP sources are connected to the backbone nodes, thus representing servers or proxies in the Internet. Every TCP source corresponds to one TCP sink. Files are generated at the sources and are sent to the respective sinks. All links are full duplex with equal capacity in both directions. The traffic characteristics in all ten access areas are assumed to be similar. We use the network simulator ns available from Berkeley Laboratories. As TCP protocol we choose the FullTcp module, which implements mechanisms for slow-start and congestion avoidance. Furthermore, the Nagle algorithm is supported

As an example, we generate an average bit rate of about 94 kbps per access area (i.e. on each access links) resulting in 940 kbps on the aggregation link. The traffic is flowing from the backbone node to the aggregation node and then traverses the respective access link. The file size is Pareto distributed with shape parameter 1.5 and mean value 12 kbytes. Access speeds (r_{peak}) are assumed to be 64 kbps. For a file size of 100 kbytes (including TCP/IP overhead) we demand an average transaction time of at most 15 seconds. Since at 64 kbps access speed the minimum transfer time of such a file is 12.5 seconds, the delay factor should be below or equal to 1.2. Applying the suggested dimensioning procedure with M/G/R PS, we obtain capacity values of 192 kbps for access links and 1.088 Mbps for the aggregation link.

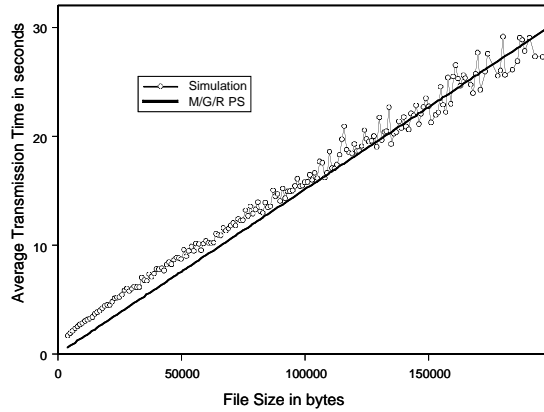


Figure 3 Transaction time vs. file size for $RTT = 30\text{ ms}$

Figure 3 illustrates a typical simulation result for a round-trip time of 30 ms. The average file transaction times are plotted over the file size x . In addition, the theoretical values according to Equation 2 are given. As we can see, small files experience a somewhat larger average transaction time than computed by Equation 2. For increasing file sizes the theoretical model becomes more accurate and, finally, even overestimates the transaction time.

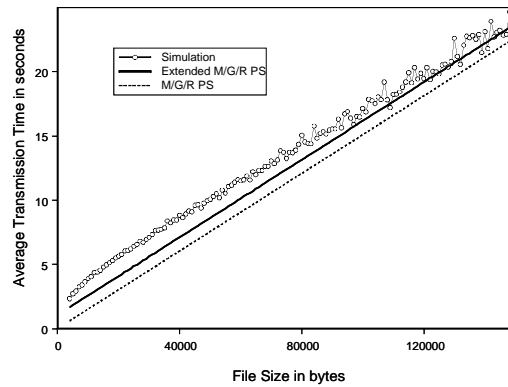


Figure 4 Transaction time vs. file size for $RTT = 300\text{ ms}$

For larger RTT , the efficiency of a TCP flow to utilize its capacity share decreases. As a consequence, the difference between measured transmission times and values computed by the pure M/G/R PS model increases, as shown in Figure 4 for $RTT = 300\text{ ms}$. Even the extended model still underestimates the transmission times of smaller files. However, for larger file lengths (≥ 100000 bytes) it comes very close to the simulated values, thus, proving it

to be a suitable tool for dimensioning access networks with elastic traffic. Similar results were obtained for scenarios with different access speeds and varying propagation delays between backbone node and servers.

5 Conclusion

In this paper we have presented a model that can be used for dimensioning star-structured access networks for elastic traffic. The dimensioning procedure is based on the M/G/R processor sharing model, which is extended to capture TCP specific mechanisms. Simulations have shown its applicability for well-defined scenarios with different access speeds and varying link delays or round-trip times. Now, further work needs to be done to investigate the performance of the presented model for TCP traffic in a more heterogeneous environment with strongly differing round-trip delays for individual servers, different access speeds per user, and different traffic characteristics. Furthermore, different notions of TCP implementations have to be considered.

References

- [1] S. Borst, O. Boxma, P. Jelenkovic, Generalized Processor Sharing with Long-Tailed Traffic Sources, ITC 16, Edinburgh, 1999
- [2] J.W. Cohen, The Multiple Phase Service Network with Generalized Processor Sharing, Acta Informatica 12, 1979
- [3] L. Kleinrock, Queueing Systems Volume II: Computer Applications, Wiley-Interscience Publication, 1976
- [4] K. Lindberger, Balancing Quality of Service, Pricing and Utilisation in Multiservice Networks with Stream and Elastic Traffic, ITC 16, Edinburgh, 1999
- [5] R. Núñez Queija, J.L. van den Berg, M.R.H. Mandjes, Performance evaluation of strategies for integration of elastic and stream traffic, Centrum voor Wiskunde en Informatica (CWI), PNA-R9903 February 28, 1999
- [6] W. R. Stevens, TCP/IP Illustrated, Volume 1: The Protocols, Addison-Wesley, 1994
- [7] A.P. Zwart, O.J. Boxma, Sojourn time asymptotics in the M/G/1 processor sharing queue, Centrum voor Wiskunde en Informatica (CWI), PNA-R9802 April 30, 1998