The Gaussian Nature of TCP in Large Networks

RESEARCH THESIS

In Partial Fulfillment of the Requirements for the Degree of Master of Science in Electrical Engineering

MARK SHIFRIN

Submitted to the Senate of the Technion - Israel Institute of Technology
Av-Alul, 5767 August 2007

The research thesis was done under the supervision of Dr. Isaac Keslassy in the Faculty of Electrical Engineering.

Acknowledgements:

Thanks to Dr. Isaac Keslassy for his energetic guidance and inspiration.
Thanks to prof. Israel Cidon and prof. Reuven Cohen for their valuable remarks.
Thanks to the Comnet lab staff: Hai Vortman, Yoram Or-Chen, Yoram Yihie and Alex for their exeptional technical support and for providing me with all the needed resources.
Thanks to Alex Shpiner and Itamar Cohen for their tips in writing this document.

Contents

List of Figures

Abstract

Most of the traffic in Internet networks belongs to TCP flows. The Internet network seems to be too complex to be fully analyzed. Feedback mechanisms of TCP, numerous protocols and complex flow interactions, together with a variety of different topologies, imply a complex statistical problem. In fact, currently there is no complete statistical network model.

In this work we present a novel approach to the analysis of a network in which most of the traffic consists of TCP flows. We find the traffic distribution of all network components, which are centralized around a bottleneck. By exploring deeply the nature of the packet losses, we find that they are bursty and correlated. This insight provides us with a novel model for the TCP congestion window, leading to statistical models of single links, and finally escalating to the level of distributions of the largest components. We utilize the achieved models to construct the pattern of the arrival rate to queues, and derive the distribution of the bottleneck queue size and a resulting loss rate for the network.

We end up with the first complete statistical description of the entire network. Further we prove that the number of packets on most network link sections follows a Gaussian distribution, and analyze as well the parameters of this distribution, thus providing a key insight into the general statistical behavior of Internet traffic.

Abbreviations

$cwnd^i$	TCP Congestion window size. Upper index i means the window belongs to the flow i
w	Same as cwnd
W	Sum of all $cwnd^i$
l_k^i	Traffic which belongs to the flow i on Line k
bcwnd	Distribution of the burst of packets traveling on the Lines after passing the buffer
L_i	Total traffic on Line i
r^i	Transmission rate of flow i
R	Total transmission rate of all flows
$G/D/1/K$	Queue with Generally distributed arrival, Deterministic service distribution, 1 active buffer with space for K packets.
$l(n)$	Distribution of the number of lost packets in window of size n with at least one loss.
$p_e(n)$	Effective packet loss for window of size n

Chapter 1

Introduction

In this section we will introduce the Large Network that we want to study, and will define its basic components. Next we will discuss our goals towards the large network analysis and what are the problems and difficulties which interfered to do it so far. We give then a comprehensive list of the related work on the TCP congestion control and buffering in large networks. Finally we present shortly the innovations and main contributions in our work, and redefine the problem as a dumbbell topology problem.

1.1 Large Networks

We refer primarily to the general case of a large topology, with a large number of TCP flows consuming a group of bottleneck links. We may see this topology as a group of independent instances of multiple sources and multiple destinations together with many routers. Figure 1.1 illustrates the suggested scenario.

Analyzing large networks, nowadays, is a highly challenging task, catching a large amount of interest due to the constantly growing dimensions of Internet network deployment. Talking about traffic in large networks, we refer mostly to the TCP data stream. In modern networks most of the traffic is TCP, while the percentage may vary from $80\% - 85\%$ at minimum up to 98% at maximum, depending on the network conditions [23][21]. A thorough network understanding is needed in order to be able to plan the requirements for the backbone routers.

The complexity of the connections and routers which monitor a millions of flows, implies several problems. Numerous sites of congestions arising as a result of slow links in some

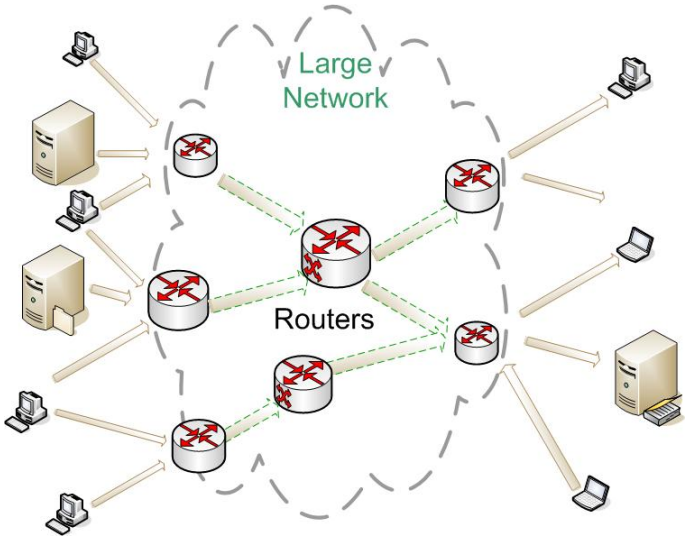

Figure 1.1: *General Case Topology Multiple sources contacting a multiple destinations. Some routers inside the network which is presented by cloud may be a bottlenecks*

routers may slow up the routing from the large group of sources to the corresponding large group of destinations. In addition a limited buffer can start abruptly dropping packets of many flows thus causing a sudden slow-down for these flows.

Research of such a network implies two main objectives: First we would like to develop the probabilistic models which will describe the statistical behavior of the network and its components. We would like to understand, using these models, what are the reasons of congestions and of high packet loss rate.
Second - we would like to obtain a planning ability. For example, having a demand for a given packet loss rate and being constrained by a limited link capacity, we would like to be able to plan the buffer size that will answer these demands. In contrary, the buffer size may be fixed and we intend to plan according to the tradeoff of packet loss and link capacity of a bottleneck.

1.2 Obstacles and Difficulties

Nowadays, to the best of our knowledge, there is no work that gives a comprehensive description of large networks. We will nominate the main reasons to the absence of such a model.

According to the TCP algorithm [24], every flow maintains a dynamic congestion window, which represents the allowed number of unacknowledged packets in the network. A group of algorithms, namely Slow Start, Congestion Avoidance, Fast Retransmit and Fast Recovery were designed to fit this window to the congestion in the network, on the path that is traversed by the data packets, according to the accepted acknowledgements.
The complicated TCP behavior and TCP feedback property are serious problems, especially if we are about to analyze the congestion window ($cwnd$) distribution. The diversity of algorithms (Slow Start, Congestion Avoidance, Fast Retransmit, Fast Recovery) introduced by Newreno implies a complex cwnd behavior.

A decisive obstacle is formed by the statistical dependency of various connections to the same router, sharing a same bottleneck. A correlation may be observed between the traffic belonging to the different flows, and between their congestion windows as well.

The research of the distribution of the Queue(Q) might be involved with a complex queueing analysis. The serving rate is deterministic, but the input rate is a function of the arrival rate probability distribution function. This implies a complex G/D/1/K analysis.

The large network might be difficult to simplify to a simple pattern such as dumbbell topology, for instance. Multiple bottlenecks might be involved, different link capacities for the different flows are potentially complicating aspects as well. Flexible topologies - flows can join and leave frequently. Some flows can change their routing paths. As a special problem we would like to mention the problem of lost acks, as well as the acks that are routed on different paths.

As we mentioned - most of the traffic is TCP, but still a small percentage of other protocols exists: UDP, ICMP, Short temporary TCP flows and more - all of these contribute as well to the total complexity.

1.3 Dumbbell Topology

We would like to introduce an approach to simplify the large network to a specific case. We present here as well several assumptions and a justification for using such a simplification. For the purpose of analysis we introduce a simplified pattern of our model as it appears on Figure 1.2. A single bottleneck imposes a specific behavior on all the connections which route their data streams through it. A limited buffer and a slow link after it dictate the performance of all the involved TCP flows.
There is a difficulty in dividing the entire large network into separate components, because

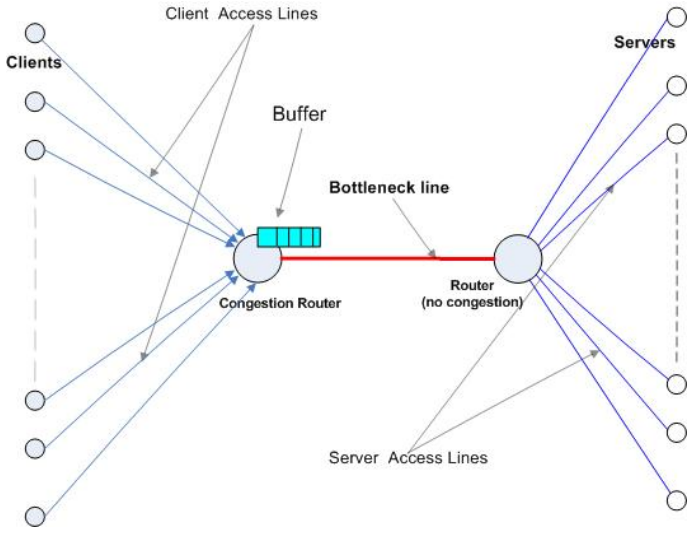

Figure 1.2: *Simplified scenario of a dumbbell topology*

of the strong interrelationships and variability of the global networks. In order to simplify the pattern of the topology we present the following assumption regarding single flows:

Assumption 1 *Every flow has only one bottleneck throughout its path.*

This assumption relies on the observation that few flows practically have more than one bottleneck, and that flows having more than one bottleneck actually mainly depend on the most congested one. We believe that dividing a large network into several sets of dumbbell topologies reflects quite reliably the real world, due to the existence of the bottlenecks in particular spots like backbone routers and main routers in networks. A dumbbell topology was also used in buffering considerations in [5],[11]. [15],[22] are additional examples of sources which utilize a dumbbell topology for the TCP analysis.
If so, we refer to the scenario based on the following assumption:

Assumption 2 *All clients have a unique common bottleneck.*

All the packets have to pass through this same link with limited capacity, denoted from now on - *forward bottleneck*. Consider a large number of clients, each one connected with its own link to the router, denoted from now on - *client forward access links.* The router has to redirect the packets to the servers, using the slow link first.

In addition consider the *client backward access links* and the *backward bottleneck*. The backward bottleneck line may not have the same capacity limit as the forward bottleneck,

as we primarily discuss the congestion only in the forward direction, **that is, there is no actual bottleneck on the backward bottleneck line.**
We assume that large networks may be reduced to a dumbell topology scale and perform our analysis on the presented dumbbell topology, working separately on its components.

Next, we would like to clear out several assumptions which we will utilize later, regarding the dumbbell topology we will use.

Assumption 3 *Link capacities of the client forward & backward access links are unlimited, i.e. very large comparatively to the forward bottleneck.*

Based on previously stated Assumptions 1 and 2 we state as well:

Assumption 4 *We assume that the only queue size limit encountered in the topology is before the forward bottleneck. Other queues are assumed to be unlimited, therefore we expect the packet loss to happen only in this queue.*

Every client runs its own TCP source, while the destination is the common server or a individual servers, provided that the bottleneck is common. The described topology has a tree form therefore no routing alternatives, for simplicity, are available.

Assumption 5 *The latency of the forward & backward access links is uniformly distributed according to some known distribution.*

Acknowledgements do not necessarily arrive on the same link for each client, and our analysis is not constrained by this condition. In addition, we assume that the overwhelming part of all the flows are long provisioned TCP flows. We assume, however that a small percentage of short TCP flows (the flows that finish before they even exit a Slow Start phase for instance) or UDP flows is also present. Our analysis refers, therefore, to the long TCP flows only, in presence of a small proportion of UDP or/and short TCP flows. Next we discuss assumptions related to the losses:

Assumption 6 *The general packet loss probability* p *is modeled as constant.*

In fact, there are fluctuations in the packet loss probability if we measure over a very short time intervals, but they are quite negligible. There are some works that model the packet loss probability as variable, as is discussed in the next section, but we are convinced that in the steady state the model of the constant p is truthful enough.

Assumption 7 *No acks are lost throughout the TCP connections - the only losses are caused by data segments lost.*

There are 3 main reasons for no acks loss: First the acks are of very small size (40 byte) and their probability to get lost is low. Second the accumulative method of acknowledging almost cancels the influence of the losses. In addition the bottleneck is normally present only in one direction. Research on the lost acks was done in [16].

1.4 Related work

Only a few sources in the literature try to give a full picture of the network, while most of the work is concentrating on the buffer occupancy distribution or buffer sizes comparisons. Other works are dealing just with the packet loss derivation using a fixed sets of assumptions. A significant amount of work dealing particularly with the *cwnd* parameter distribution is also present. For simplicity we divide our comparison according to the topics.

1.4.1 *cwnd* parameter - distribution and statistical models

Note that we sometimes refer to the *cwnd* as w - a window distribution of the single TCP process. Most of the works deal mainly with the moments calculation of w, particularly the mean and the variance [21][1][20]. The main contrast between our work and the other works on *cwnd* is that we strongly rely on the correlated nature of the losses, while others assume independency. Previous attempts to emphasize the bursty nature of losses were done on [3],[12]. We present in our work a full description of the distribution of the correlated losses.

Reference [14] performs both a discrete and a fluid model, and makes a comparison between them. The discrete model is performed by SMP (Semi Markov Process). For the fluid model there is an assumption of Poisson arrival. The losses are uncorrelated, an assumption with which we argue. Another work that utilizes SMP is [7]. The pdf in [2] is derived using differential equations. The packet loss p is independent for any packet. It is assumed that only one packet can be lost for any window. The packet loss, however, is modeled as depending on the window size. The solution is derived by solving a stochastic differential equation. The CDF rather than the PDF is derived in [18]. Authors present 3 loss models (correlated, not correlated, partially correlated). These models are implemented in their own simulator and then compared to the NS2 result. The losses of the

acks are addressed as well.

There are models as well that perform derivation for the idealized TCP, with the congestion avoidance only. In [19] the packet losses are independent events with equal probability. W is not limited, and we demonstrate in our work that when this limit is relaxed, the model is not suitable for small p. The packet arrival is presented as a Poisson process, the loss process is also presented as a Poisson process with another mean. Using the definition of these two processes the distribution and mean and variance of w is derived, as a function of the ratio of those two Poisson processes.

The model in [9] is based on a very simple Markov chain, without any correlations between successive loss events being considered, then any lost event is just a single packet loss. The model is simplified by omitting the Timeout (TO). Our model is based on a simple Markov chain as well, but we present a more precise result, based on more accurate transition probabilities. Also, as we mentioned, it is incorporating correlated losses. There is an approximation for the case when $p \to 0$ in this work, for which the results are more precise.

1.4.2 Works Dealing with Buffer Sizing and Network Considerations

We would like to mention as well numerous works which we found as most interesting, dealing with the buffer sizing and packet loss derivation. Our work deals mostly with small buffers, and we derive the packet loss, which is a function of the buffer size as well. We treat in the conclusions some aspects regarding large buffers as well.

First we would like to refer to the rule-of-thumb presented in [28]. The main objective in buffer sizing was to keep the link always busy and avoid any unexploited link. The rule of thumb states that the buffer size is given by $B = C \times RTT$, where C is the bottleneck rate and RTT is the round trip time. This rule of thumb implies very large and slow buffers, typically using a slow DRAM and inflicting a great queueing delay on all the traversing flows.

Our major inspiration is given by [5]. This article proposes the Stanford Model - Gaussian model for the total W (sum of *cwnd* of all flows sharing the same bottleneck), and for the Q (the size of the queue in the buffer before the bottleneck) as well. As we mentioned, we gave a slight emphasis to the smaller buffers, and we couldn't see the Gaussian distribution in the queue. However, we observed the Gaussian distribution not only on the W but also on the various links groups in the topology. We confirm that the condition for the Gaussian distribution is desynchronization among the flows.

We assume, as well that the buffer is small enough, or even smaller than in the Stanford Model. Using the bottleneck link capacity (C), it is expressed as $B \approx C \times RTT/\sqrt{N}/\alpha$, for a number of flows equal to N, then α can be up to 20.

In [4] the packet loss is derived using the $M/M/1/K$ model, as a function of load, while the load is derived by summing throughputs of the flows, and dividing by the link capacity C. Finally the closed form formulae for calculating the loss probability p given buffer size are derived. Unlike our approach, the analysis uses Poisson arrival.

In [10], in contrary, no Poisson arrival is put in use, but a "paced" pattern of the arrival is suggested. The pacing phenomenon is said to correctly model networks with small buffers and slow access lines. The buffer as small as in the order of $O(log(W_{max}))$, where W_{max} is the maximum permitted *cwnd*, still can suffice to provide tolerable packet loss. For overprovisioned networks the buffer size is given as a function of the desired load θ and the overprovision factor ρ. For the paced scenario the packet loss is proved to be lower than $O(1/W_{max}^2)$. These results are theoretical basis for the described scenario, while we present results that are good for underprovisioned network, i.e. the bottleneck link is below its maximum capacity, and the packet arrival is general, but bursty (no Poisson assumption yet), ruled solely by the TCP dynamics. In addition, this work completely relaxes the dependence of the buffer size on the bandwidth-delay product. Quite similar assumptions and conclusions to [10] are brought in [6].

An approach developed in [8] allows to find B according to two alternative constraints. First is the desired load - it is proven, that for a load of almost 100% a buffer of just a tenth of the rule-of-thumb can suffice to generate a packet loss of less than 5%. We show in our model that for w with packet loss that high, the flows can evolve their *cwnd* up to no more then 5-6 packets, or even less. The second is the desired packet loss. The number of the lost packets in a congestion event is observed to be a linear function of the number of flows. The slope of the linear dependency is some α which is found through the simulations. The result is good however for 5 to 200 flows, for more then 200 flows it stops to behave linearly.

In [25] a Poisson arrival is assumed again. It is stated that this assumption is actually not realistic but for short timescales is still acceptable. Therefore, the loss probability can be calculated from a Markov chain model, $M/M/1/K$. Instability is discussed: the instability, or the size of the oscillations in the buffers instantaneous occupancy arises with the synchronization of the flows. For the small buffers - increasing the size of the buffer causes instability - large oscillations. Too small buffers lower the utilization. The buffer size can be found at the intersection of the equation of the loss probability as a function

of the load, and the differential equation of the load, at the equilibrium point. We do not treat in our work the stability problem.

The analysis of the M/D/1 approximation used for the packet loss equation as a function of the load in [26]. It is suggested, that a small buffer actually *promotes* desynchronization of the flows. Actually we can give a strong confirmation to this conclusion, as we observed it clearly in our NS2 [30] simulations.

1.5 Major Contributions

We present in our work a novel approach of analyzing the entire network, in which most of the traffic is consumed by TCP connections. As we mentioned, we simplify the large network to the more tractable scenario of the single bottleneck, represented as dumbbell topology.

Our approach to derive a packet loss is to develop a closed loop of the models, assuming a preliminary knowledge of the loss rate and then finding the real packet loss by finding the fixed-point solution of an equation of the form $p = f(p)$. We assume, first, that we know the distribution of cwnd, and use it in order to find the traffic distribution on the lines entering the queue before the bottleneck. Using this traffic distribution on the lines we find the arrival rate on each corresponding line and consequently the total arrival rate. Next we use this total rate for the queue analysis to find the distribution of the queue occupancy and finally find the packet loss rate, using the queue distribution, finishing the closed loop.

In order to make the loop complete the model of cwnd is needed, and we find it by exploring the correlation of losses. Our objective is to find the model of cwnd knowing the packet loss probability p, in order to use this model of cwnd in the closed loop we just have presented. We explore the packet loss nature and we find that the packet loss is rather bursty. Using this burstiness we prove that the packet losses are highly correlated. This correlation leads to the conclusion that several packets might be lost during each *packet loss event*. Taking advantage of Newreno capabilities to treat several lost packets in a same Fast Recovery session, we offer a novel model for the distribution of w that treats correctly the correlated packet losses, utilizing the deep insight on queue dynamics. We develop a detailed Markov chain that allows to track very precisely the distribution of w measured with various NS2 simulations, for different packet loss levels.

Using the developed distribution of w we find the distribution of the number of packets

on every link. We prove that the total number of packets on various link sections is Gaussian distributed. We show that the total number of packets in the topology is Gaussian distributed as well, thus accomplishing the statistical modeling, which started with the single flow distribution statistics and ended with the complete statistical description of the entire network.

We define a new notion - *bcwnd* - a burst of packets that is traveling the lines after passing the queue and find its distribution. Using this *bcwnd* we find the distribution of traffic on the links in all locations of the topology, and finally find W (the total window - sum of congestion windows of all flows), ending up a first complete statistical description of the large network.

Our conclusions are that even for quite small buffers the packet loss is still tolerable, and the network performs correctly. We present numerous results for the packet loss predictions. Our analysis is unique in the sense of deriving the packet loss without using any load considerations, and also without any assumptions on the arrival probability distributions.

1.5.1 Thesis Outline

The rest of the work is organized as follows: Chapter 2 contains the closed loop for derivation of the packet loss p. Chapter 3 presents the model of cwnd and studies the correlated losses. Chapter 4 gives the models of the different components of the dumbbell topology. Chapter 5 summarizes simulations and results that check the correctness of our analysis and discusses some special cases. We finalize our work in Chapter 6, which presents conclusions.

Chapter 2

Closed Loop for the Packet Loss Rate Derivation

In this section we develop step by step a model for the packet loss rate for a network which is represented by dumbbell topology with a given set of propagation times of all flows, which are traversing a bottleneck with a limited buffer of size B followed by a link of serving rate C. We assume no additional global parameters about the network.
The closed loop is presented in Figure 2.1 We assume first that the general packet loss rate p is given, and find a distribution of the congestion window *cwnd*. Next we find the distribution of the traffic on the forward client access lines - for each flow. We progress next to the rate of each flow and to the total arrival rate to the bottleneck queue. We use the achieved distribution of the rate to construct a pdf of the queue Q and finally to compute the packet loss rate p.

The model of cwnd involves an exhaustive research of the packet loss nature and the correlation between lost packets and we postpone it to the next chapter which will be devoted for this model.

Before we start with the first phase of the loop - we would like to divide the dumbbell topology into several components, and to name all the components, as defined in Figure 2.2. We define the propagation times of the links which are located before the bottleneck, i.e. the client forward access links, as tp_1^i, where i stands for the number of the link and 1 means the enumerator of this group of lines. We denote the propagation times of the client backward access links as tp_2^i. Denote as $l_1^i(t)$ the instantaneous number of packets that present on client forward access link i, and $l_2^i(t)$ the instantaneous number of packets that present on client backward access link i. The sum of all packets on client forward

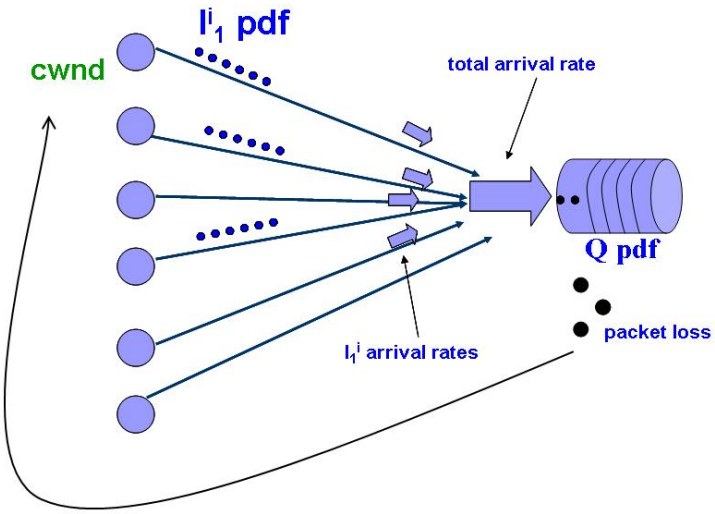

Figure 2.1: *Closed loop for finding the packet loss rate by solving a fixed point equation*

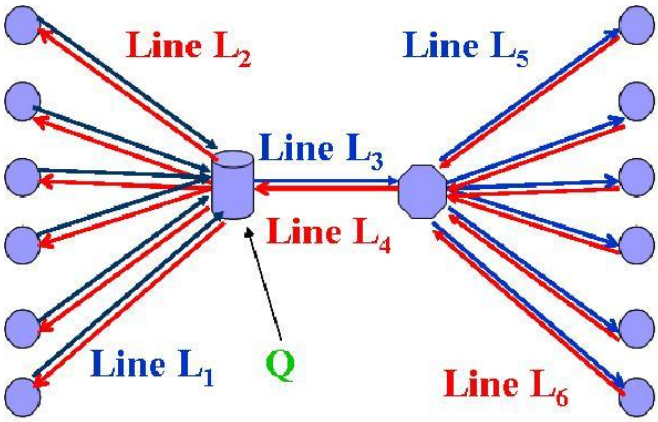

Figure 2.2: *Lines - basic components of a dumbbell topology*

1	0.066367	0.14018	0.057976	-0.02528
0.066367	1	0.05363	0.000517	-0.08138
0.14018	0.05363	1	0.062633	0.051123
0.057976	0.000517	0.062633	1	0.002982
-0.02528	-0.08138	0.051123	0.002982	1

Table 2.1: *Matrix of covariance coefficients of 5 arbitrary flows, in a scenario which included a network with 500 flows. The covariance coefficients are very far from the maximum 1.*

access links is $L_1(t)$ and its distribution is L_1. Equivalently we denote the sum of all packets on the backward client access links, bottleneck link, backward bottleneck link, and their distributions respectively by $L_2(t)$, $L_3(t)$, $L_4(t)$, and L_2, L_3, L_4. In addition to the client access links there are 2 groups of links after the bottleneck to (and from) the corresponding connection server - denote them as *server forward access links* and *server backward access links*. The corresponding definitions will be as follows: tp_5^i, l_5^i, L_5 for the forward direction and tp_6^i, l_6^i, L_6 for the backward direction.

2.1 Model of the l_i^1

We assume a given model of the TCP congestion window *cwnd* and find the model of l_1^i in this subsection. We would like to emphasize several observations about *cwnd*, which we will utilize as assumptions for our model. Denote the congestion window of each flow as $cwnd^i$.

Assumption 8 *Independence - we assume that the $cwnd^i$ of different flows are statistically independent.*

The $cwnd^i$ dynamics of each flow is controlled by its own rtt^i and the Q state. All the flows eventually contribute to the final Q state, as well as to the occupancy of the bottleneck link. However, for a steady state we can say that the pdf of the $cwnd^i$ of each flow is equally influenced by the distribution of Q, which is unique and common to all $cwnd^i$, as stated Assumption 4. We strengthen our statement by simulation results of the covariance between $cwnd^i$ of different flows, which were measured to be very low (comparatively to the variance), thus indicating a good approximation to the statistical independency. Table 2.1 presents an example of the low covariance. We need the assumption about distributions to start the model description:

Assumption 9 *We assume as well that the* $cwnd^i$ *are equally distributed.*

The packets, and also the acks, are being sent in a highly bursty manner. The service rates of the client forward access links are high (Assumption 3), and the time of sending a packet is very short comparatively to the rtt^i. Based on this fact we have the following assumption:

Assumption 10 *As a result of bursty transmissions, all the packets (or acks), belonging to the window of some flow i, are always present on the same line, i.e. either* L_1,L_2,L_3,L_4,L_5 *or* L_6.

They also may be present in the buffer, but since it is small enough, and the queueing delay is also negligible, we neglect here this probability. This assumption will not fit for special cases, when the latencies of the lines are extremely short, or when the capacities of these lines are limited to a small number of packets. Since this is not our situation, the only argument against this assumption would be that the packets should be scattered on the lines as a result of TCP dynamics. We argue against this approach as well, and show that a uniform distribution of the packets on the lines is far from the real situation and leads to an incorrect description of the traffic. The uniform distribution is also argued in [11],[12].
Before proceeding to the model of l_1^i we would like to make a short discussion about rtt and the queueing delay. For a service rate C the worst case delay of an arbitrary packet is $B_N * 1/C$ seconds for one packet, when the buffer is full. For a buffer of a size given by the Stanford Model, $B_N = C * RTT/sqrt(N)$, the queueing delay will satisfy $t_q \leq RTT/(B_N * sqrt(N)) * B_N$, where B_N is the buffer size in a topology containing N flows, answering exactly the Stanford model rule. (The maximum Queue size is equal to B_N, and this is a worst case for the t_q) Therefore, as long as we use buffers which fit the Stanford model or less, we can neglect the queuing delay and state the following assumption:

Assumption 11 *rtt equals twice the propagation time, i.e.* $rtt^i \approx 2t_p^i$, *for each flow i.*

Where t_p^i is the propagation time, i.e. the time which taken by a packet belonging to a flow i to travel from the source to the destination. We derive the distribution of the packets on l_1^i in the following theorem:

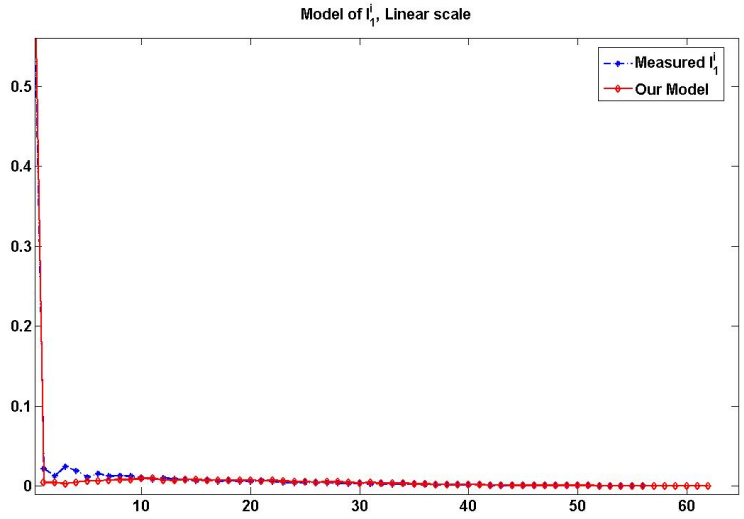

Figure 2.3: *Linear presentation of the pdf of number of packets on forward access links*

Theorem 1 *The number of packets belonging to the flow i on line L_1 is distributed as:*

$$l_1^i = \begin{cases} 0 & \text{with probability } 1 - tp_1^i/rtt^i \\ n & \text{with probability } tp_1^i/rtt^i * Pr(w = n) \end{cases} \tag{2.1.1}$$

Proof 1 *Using the Assumption 10, the probability that no packets are present on L_1 is $1 - tp_1^i/rtt^i$. The probability of presence of any number of packets is independent of the distribution of the cwnd, therefore the probability of n packet being present is $tp_1^i/rtt^i * Pr(w = n)$.*
□

We also compare the presented model with a fluid model where packets are distributed uniformly on all the lines. According to this model the number of packets present on l_1^i is given by $l_1(t)^i = w^i(t) * tp_1^i/rtt^i$. The maximum number of packets, therefore is $w_{max} \cdot tp_1^i/rtt^i$.
Figure 2.3 and 2.4 display that our model is fairly close to the measured results, throughout all the scale. There is also a presentation of comparison with the fluid model, which doesn't fit. As we see there is a positive probability that $w_{max} = 64$ packets are present on l_1^i. The alternative fluid model cannot get to w_{max} at all.

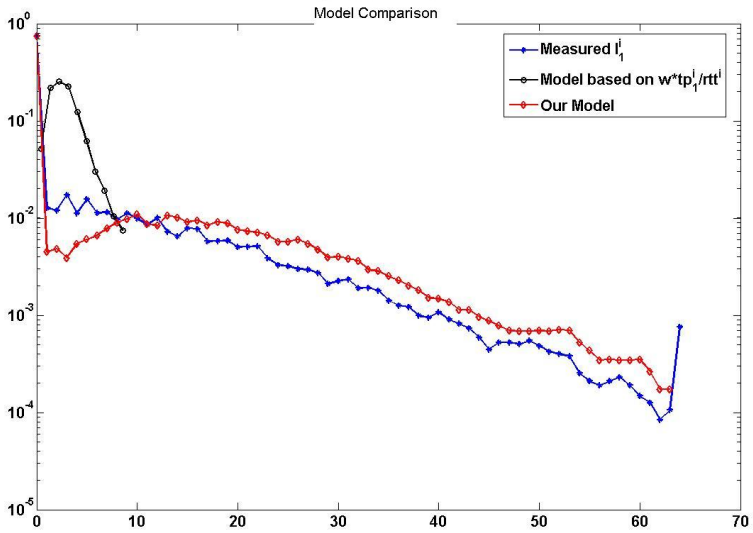

Figure 2.4: *Logarithmic presentation of the pdf of number of packets on forward access links.*

2.2 Arrival rates of single flows and total arrival rate

Denote the distribution of the arrival rate of flow i as r^i. r^i is the distribution of the number of packets arrived in δt seconds. Our objective in this section is to use the model for l_1^i to generate the model for the *rate* of each flow i, r^i, and then, using this rate distribution, to find the distribution of the total transmission rate on client forward access links. The resulting rate will lead us to the queue distribution and finally to the packet loss for the entire topology, thus accomplishing the final merge of all the probability models we introduced so far. Hence we would like to get an insight on the forward access links. We know the pdf of the instantaneous number of packets. We assumed that, taking advantage of the short transmission time, and the TCP dynamics as well, all the packets in the same window are coming in a single burst. We are interested in finding r^i, the number of packets arrived in δt seconds. The choice of δt must satisfy a condition $\delta t \gg t_{tr}$, when t_{tr} is the transmission time of the maximum size burst. The demonstration of the r^i is illustrated on Figure 2.5. The transmission is bursty and the distribution of l_1^i represents, actually, the number of packets arrived in tp_1^i seconds. To find the arrival rate in δt seconds we utilize the following additional assumption:

Assumption 12 *Given the pdf of l_1^i, the probability that on some link i $x > 0$ packets*

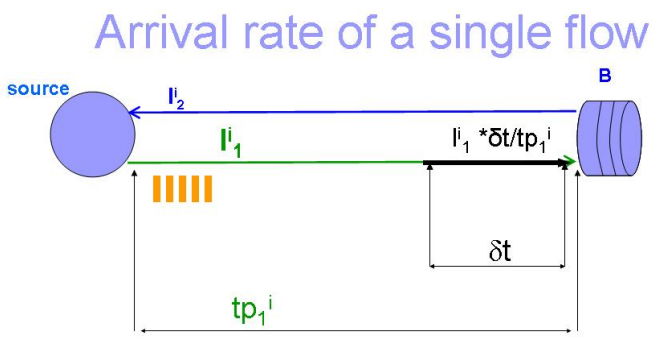

Figure 2.5: *Model of the rate - number of packets sent in δt time*

arrive within δt, such that $\delta t < tp_1^i$, is $P(l_1^i = x) * \frac{\delta t}{tp_1^i}$.

Using the Assumption 12 we can conclude that the rate arrival will be according to the l_1^i distribution, and the arrival of any burst of packets within period δt can occur with probability $\delta t/tp_1^i$. We define the pdf of the rate for the flow i, denoted as r^i in the following theorem:

Theorem 2 *The number of packets r^i of flow i arrived during $\delta t < tp_1^i$ is distributed as:*

$$Pr(r^i = x) = \begin{cases} Pr(l_1^i = x) * \frac{\delta t}{tp_1^i} & for\ 0 < x \leq 64 \\ 1 - \delta t/tp_1^i + Pr(l_1^i = 0) * \frac{\delta t}{tp_1^i} & for\ x = 0 \end{cases} \tag{2.2.2}$$

Proof 2 *Assumption 12 gives us the probability for the packet arrival of any size larger then zero. The complementary probability, therefore, stands for the no arrival event.*
□

That is, there is a packet burst arrival of any size (including the size of zero packets) with probability $\delta t/tp_1^i$, weighed by the probability of the size of the arrival, otherwise there is no arrival. There is a probability of zero arrival as well, because l_1^i can have no single packet (and in fact most of the time it is empty).

Next we assume the following assumption before finding the total arrival rate:

Assumption 13 *Independence - we assume that the r^i are statistically independent.*

Since both the $cwnd^i$ and l_1^i are statistically independent, this assumption is absolutely natural. The total arrival rate R is given by the following theorem:

Theorem 3 *For a large number of flows $N \gg 1$, the total arrival rate R from the forward access links has a Gaussian distribution:*

$R \sim Norm(\Sigma_i E(r^i), \Sigma_i var(r^i))$

Proof 3 *Using the independence stated in Assumption 13, $E(R) = \sum_n E(r^i(t))$, $var(R) = \sum_n var(r^i(t))$. The distributions of the r^i are similar but with different parameters. In order to prove the Gaussian nature of R, we use the Lindeberg Central Limit Theorem, showing that the Lindeberg condition is true (by Zabell [29]).*

Denote as μ_i the expected value of the arrival rate on the client forward access link $E(r^i)$, and denote as σ_i its standard deviation. We define the following sum : $s_N^2 = \sum_{i=1}^N \sigma_i^2$. Then, for every $\varepsilon > 0$ we want to prove the following condition:

$$\lim_{N\to\infty} \sum_{i=1}^{N} E(\frac{(r^i - \mu_i)^2}{s_N^2} : |r^i - \mu_i| > \varepsilon \cdot s_N) = 0 \tag{2.2.3}$$

where $E(U : V > c)$ is $E(U1\{V > c\})$, i.e., the expectation of the random variable $U1\{V > c\}$ whose value is U if $V > c$ and zero otherwise. This is the Lindeberg condition for the Lindeberg Central Limit Theorem. We can rewrite the condition as follows:

$$\lim_{N\to\infty} \sum_{i=1}^{N} E(\frac{(r^i - \mu_i)^2}{s_N^2} \cdot 1\{|r^i - \mu_i| > \varepsilon \cdot s_N\}) = 0 \tag{2.2.4}$$

The meaning of this condition is that there is no capture phenomena, in which a unique flow or a small share of all the flows seize the major part of the link capacity.

We proceed with several statements about the mean of the arrival rate of each TCP flow and its bound. The mean of the r^i, according to its pdf given in Theorem 2 is as follows:

$$E[r^i] = \mu_i = 0 * (1 - \frac{\delta t}{tp_1^i}) + \frac{\delta t}{tp_1^i} * \sum Pr(l_1^i = n) * n \tag{2.2.5}$$

where rtt^i is approximately equal to $\sum_{k=1}^{6} tp_k^i$. The propagation times of all the links tp_1^i are limited and chosen out of some uniform distribution within positive bounds. Hence, the expected value of the r^i is bounded as follows:

$$\frac{\delta t}{max(tp_1^i)} \leq \mu_i \leq \frac{\delta t}{min(tp_1^i)} * w_{max} \tag{2.2.6}$$

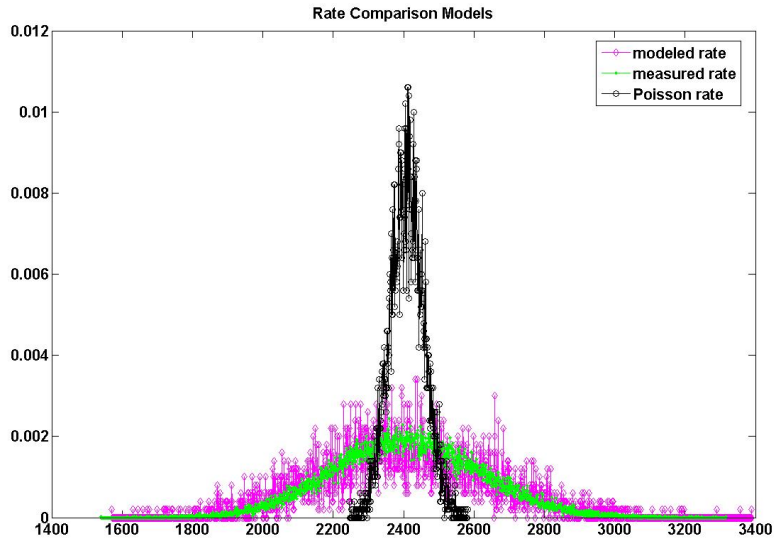

Figure 2.6: *Model of rate for* 2000 *flows. The horizontal axis stands for the total number of packets transmitted in* δt *time. The vertical axis is the probability scale.*

where w_{max} *is* predefined *TCP maximum cwnd (awnd), and normally is equal to* 64*. The value of* $|r^i - \mu_i|$ *is bounded as well. This follows from the bound on the value of* r^i*, which is limited in the range* 0 *to* $w_{max} = 64$*. Thus* $|r^i - \mu_i| \leq w_{max}$*.*

Next, we find for every ε *some* N_0*, such that for* $N \geq N_0$*,* s_N *will fulfill the condition* $s_N \geq w_{max}/\varepsilon$*, where* w_{max}*, as defined, is the maximum window limit imposed by the TCP. This* N_0 *exists because* s_N^2 *grows linearly with N. Then we have:*

$$Pr(|r^i - \mu_i| > \varepsilon \cdot s_N) \leq Pr(|r^i - \mu_i| > \varepsilon \cdot \frac{W_{max}}{\varepsilon}) = 0 \tag{2.2.7}$$

Using the last equation we can find for any ε *such* N_0 *, which will satisfy the Lindeberg condition.* □

We showed that the transmission rate is Gaussian. We would like to verify if we could get a better model, by using a Poisson distribution. The Poisson model with the same mean gives very small variance and - see Figure 2.6

Additional results for the rate are introduced in the Chapter 5.

2.3 Queue distribution and Packet Loss Derivation

In this subsection we find the Q distribution and the general packet loss approximation p. We suggest two ways to find the Q distribution. Having the pdf of the total arrival rate R, we can easily derive the packet loss of the entire topology, provided we know B - the bottleneck buffer size and C - the bottleneck capacity. The derivation may be done numerically, for example by a simple $MATLAB$ simulation of the Markov chain of the Q, with R as the distribution of the arrival and a deterministic service of rate C. For the simulation we produce a sufficiently long vector and filter it through the Q, dropping the packets when $Q = B$.

An alternative analytical way to find the Q distribution is using the G/D/1/K analysis, in which we use the rate distribution R we just found in place of a generally distributed input. We implemented the algorithm used in [27], we briefly describe this algorithm in Appendix A. The results are presented on Figure 2.7. We compare 4 graphs in this figure. The measured graph has a smooth edge when Q is close to B and to 0. We could not reproduce such a smooth behavior because of the burstiness assumption and discrete analysis. We assumed that all the packets arrive in the same burst simultaneously, however in fact there is a very short range between the arrivals of the sequential packets. An additional confirmation for this conclusion is provided by the graph of the exact measured arrival rate, which was simulated with the Markov Chain. We can see exactly the same type of discrepancy in this case. It is worth mentioning that we found the distribution of the Q using the models of l_1^i, r^i and R. Thus, we performed already the major part of the closed loop.
We should expect therefore, as is seen from the graph, that the packet loss will be somewhat higher in our model than in the measured results. We confirm this when we introduce the final results for p.

2.3.1 Derivation of the packet loss rate p

We continue with the derivation of the packet loss rate p. The distribution of Q gives us a straightforward way to compute p, as long as it is constant (using Assumption 6). The loop of models $p \Rightarrow cwnd \Rightarrow l_1^i \Rightarrow r^i \Rightarrow R \Rightarrow Q$ can be written actually as expression $f(p)$. Because of the last relation $Q \Rightarrow p$, holds $p = f(p)$. (We didn't show yet the first part of this loop as we postpone it to the next chapter.) We find p by solving the equation $p = f(p)$, using a gradient descent algorithm. We use first some p for finding the cwnd, then we perform all the distribution models as were described in this section

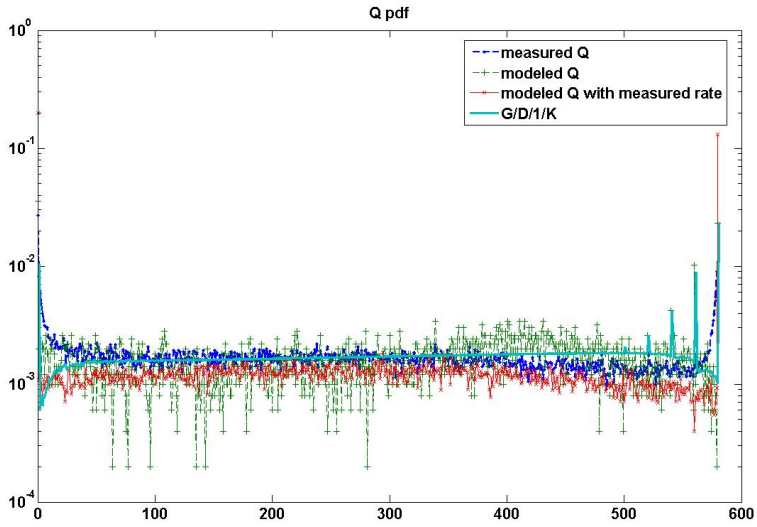

Figure 2.7: *Example of Q distribution for 2000 flows with Buffer of 580 packets The four pdf lines represent the measured Q by NS2 simulation, the model of the Q, using the modeled rate, then the modeled Q, using the measured rate, and finally the $G/D/1/K$ result. The drawbacks of the discrete analysis are clearly seen on the edges of the Q distribution. We also model the Q with the rate measured by NS2, to stand this drawback*

Measured	Model
2.70%	3%
1.40%	1.82%
0.80%	0.98%
0.45%	0.56%

Table 2.2: *Packet loss results, solved by finding fixed point of p=f(p). The packet loss gives an upper bound of the simulated value.*

and get some new p out of Q distribution. We make a correction against the gradient of the error, multiplying the error by some "*step*", and adding the product to the old p. We repeat this process, till the error is small enough. The results for several cases are summarized in Table 2.2. We performed the fixed point solutions using the closed loop of the statistical models, and the results as expected give an upper bound which is quite tight, comparatively to the measured values.

Once we have found p the closed loop of statistical models is finished. We have now the ability to find the packet loss of a topology with given rtt distribution, buffer size and bottleneck link service rate. We used a distribution of cwnd in this section, and in next chapter we show how we find this model, exploring the nature of correlated losses.

2.3.2 Complexity of the Calculation

The calculation of p demands running the gradient descent algorithm for several iterations. The convergence may be achieved through typically less then 50 iterations in case the original assumption was far from the fixed point. In case it was close the convergence is very quick and can take less than 10 iterations. The exact number of needed iterations is depends on the desired precision.
Next, we are about to discuss the complexity of a single iteration.

1. cwnd calculation - constant complexity: O(1)
2. l_1^i models for N flows: O(N)
3. r^i models of the flows: O(N)
4. model of R - the total rate: O(1)

5. Calculating the pdf of Q and the final packet loss: O(I), where I represents the complexity of calculation of the pdf of Q. O(I) depends on the approach used, and is polynomial in B.

6. p correction: O(1)

For a small number of flows, the complexity of finding the fixed point is *independent on the number of flows N*, and depends on the desired precision of p. We performed our solution for the approach of calculating the Q with a Markov Chain simulation with vector R. The length of the simulation determines the level of precision. The process may be done by building the transition matrix of the Markov Chain of Q and then finding the steady states. The size of this matrix is $B \times B$. The solution which utilizes a G/D/1/K analysis depends on the size of the buffer, because the convergence of Q is iterative and depends on the number of possible states. Therefore, in both approaches, O(I) is dependent of B and the desired precision level. For the cases of millions flows $O(N) \gg O(I)$ and the final complexity would be approximated as O(2N).
In summary the final complexity may be bound for any case of topology by O(2N+I).

Chapter 3

Model of congestion window distribution

The distribution of the cwnd, i.e. the steady state pdf of the TCP congestion window size, has its special importance. To the best of our knowledge only a handful sources in the literature derived a full distribution, while most of the sources concentrated on its average and the average throughput calculation. The precise model for the w distribution is needed in order to derive the distribution of the number of packets on forward access links l_1^i, as we saw in the previous section. We start this section with a research of the correlated losses. We show and prove that the distribution of the lossy bursts is identical for any network condition. We use this distribution to define the *effective packet loss*, and to build a simple Markov Chain for the cwnd, by utilizing the property of Newreno that during the Fast Recovery multiple packets can be recovered, without additional window reduction. Next, we introduce a few possible refinements for this Markov Chain, including an approximation of Slow Start. Finally we bring a comparison of our model to the NS2 simulated results, for the different packet loss rates.

3.1 Correlated Losses

The basic idea we promote in this section is the fact that losses in a window are correlated. As a consequence, in any window with losses, the probability that more than one packet was lost is not negligible. This statement stands in opposite to the models from the literature with uncorrelated losses - in which the probability of every packet to fall is independent and equal to p , and so the probability for 2 or more packets to be lost is

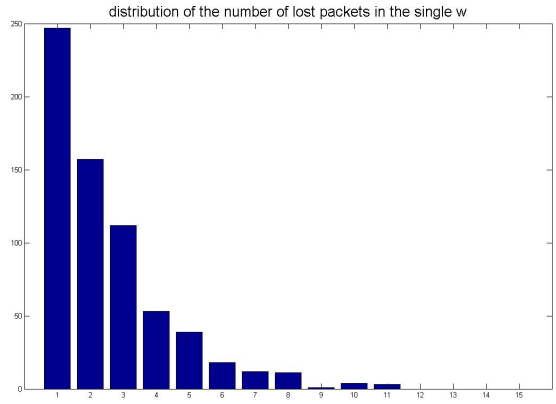

Figure 3.1: *Distribution of the number of lost packets in a single window, given that at least one packet was lost in this window. This histogram brings results for all possible sizes of windows in common.*

very small and negligible.
Therefore, unlike most of the models we saw before, our model is strongly based on the correlated losses. We measured, using NS2, the distribution of the number of lost packets, given the loss already happened in a single window. It can be seen from Figure 3.1 that in less than half of the cases the number of lost packets was just 1. In other cases 2, 3 or more packets were lost. We observe that this distribution is not dependent on the packet loss probability p, and once the *loss event* happened - the same probability for 1,2,3 or more packets to be lost is observed, independently of the general packet loss rate in the network.

It is worth mentioning, that by now we didn't take into consideration the window sizes, and we demonstrated results for all possible windows in common. It is clear that for a window of size 3 no more than 3 packets could fall so the correlation is even stronger when it follows from Figure 3.1.

Let's discuss the distribution of the losses for the different window sizes. Using again NS2 simulations, we researched separately the loss distribution for different window sizes. The results are presented on Figure 3.2 and Figure 3.3. These results encourage us to form the following assumption:

Assumption 14 *The distribution of the number of lost packets (lost burst) in a window of size n, given that at least one packet was lost, is independent of the general network packet loss, and independent of other network parameters (buffer size, link capacity, RTT*

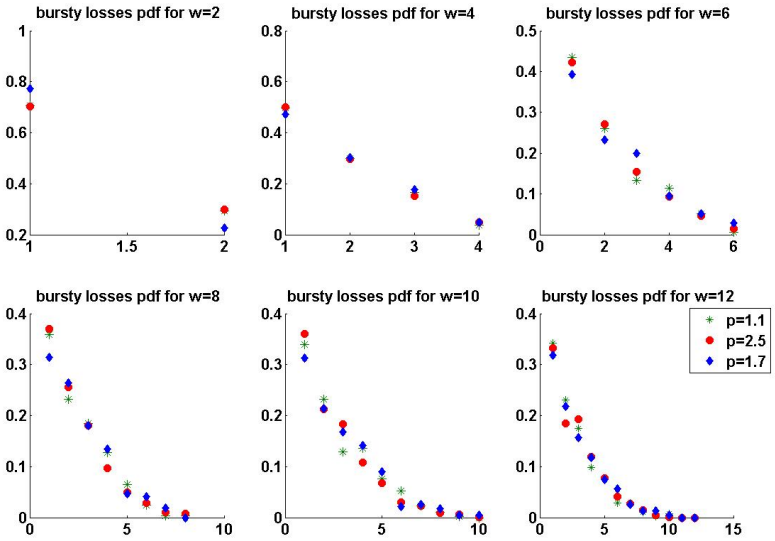

Figure 3.2: *Distribution of the number of lost packets according to the window size, given that at least one packet was lost in this window. This figure brings results separately for the window sizes 2,8,6,10,12,14. 3 cases of p presented here. For all 3 different packet loss cases shown here, the distribution is the same. (*NS2 simulated results*)*

distribution).

In another words, p influences both the probability of a loss in a window, and the window size distribution; but given both of these (i.e., given that there was at least one loss in a window of some given size n), the distribution of the number of losses in the window is fixed. If so, once there was a packet loss event in the window - the distribution of the number of the lost packets for the same window size will be the same. This is a strong assumption, which we confirmed by NS2 simulations. We discuss it more in detail, next, towards a possible explanation.

We provide some intuition next for the distribution we saw. Q is changing and balancing and reaches its peak from time to time, approximately periodically, and it stays in the peak for some period, where it fluctuates from B to $B - b$, where b is some number of packets variating from 1 to nearly 3. Most of the time $b = 1$. The losses can occur only when Q is at its peak period. For the high packet loss these peak periods last longer, and happen more often, but once the peak period occurred, the mutual dynamics of a packet arrival and Q fluctuations are approximately the same. Particularly, if we compute the ratio of the arrival to the frequency of the fluctuations of Q - we will see approximately

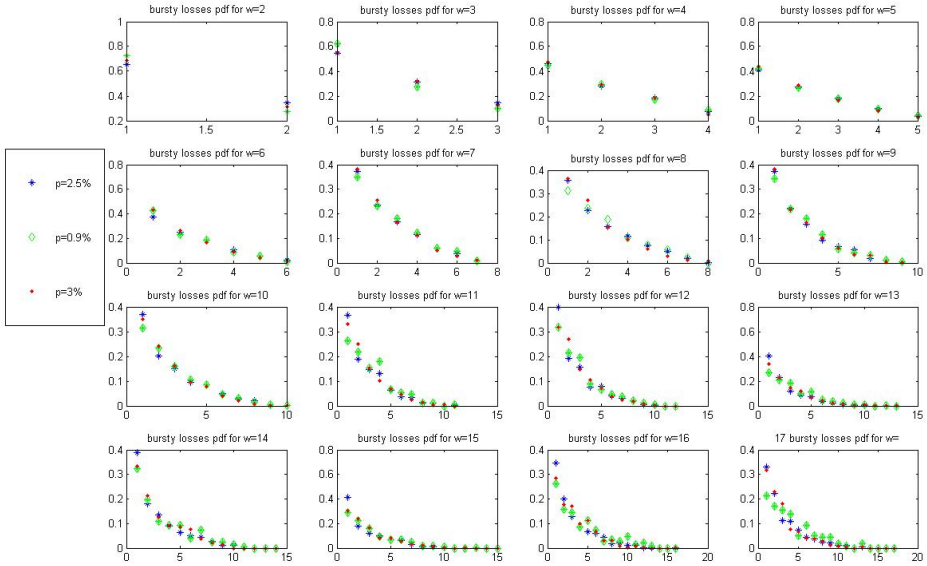

Figure 3.3: *Distribution of the number of lost packets according to the window size, given that at least one packet was lost in this window. This Figure is the same as Figure 3.2, but with more examples of window sizes (window sizes 2 to 17), and with different overall packet loss rates. (NS2 simulated results)*

the same ratio for the different p and network conditions. The example could be seen in Figure 3.4.

We observed the same behavior of the Queue for buffers of size which was more then Stanford model size, and a distribution of the bursty losses holds for quite large buffers, covering the range of small and medium (Stanford Model size) buffers at least. Quite similar fluctuation could be spotted as well in even larger buffers, which sizes approaching a rule-of-thumb size. We may expect, therefore the same distribution of the correlated losses even for the large buffers.

3.2 The Packet Loss Event probability

We define the p_e - the Packet Loss Event probability per packet - as a per-packet probability that drops (no matter how many) happened in a window, i.e. the probability this window experienced a loss event, per packet. The probability of the Loss Event for window size n, then, is equal to $p_e * n$. $p_e(n)$ refers to the losses for window size $w = n$. We do not discriminate the packets according to their location in the window. This leads us to the additional assumption:

Assumption 15 *$p_e(n)$ is equal for every transmitted packet in the same window of size n.*

$p_e(n)$ is the *effective packet loss*, which is independent for every packet, in window of size n. Intuition: in another words, we effectively use the scenario with uncorrelated packet losses, with packet loss equal to p_e. The reason for such an interpretation is the quality of Newreno, which is able, using the Fast Retransmit and Fast Recovery , to recover from several lost packets in the same window almost at the same price as if only one packet was lost. In order to justify the last statement we utilize the following assumption:

Assumption 16 *The recovery process from any number of losses takes one RTT.*

We need this assumption in order to build a simple Markov Chain. Obviously this assumption doesn't reflect the real situation, because Fast Recovery may last for several RTT. However, in steady state it proves itself as justifiable because it leads us to a fairly precise final model of *cwnd*.
Clearly, $p_e(n) < p$, for all $n > 1$, where p is the original packet loss, observed in the network, per single packet, which is constant in the steady state (Assumption 6). We

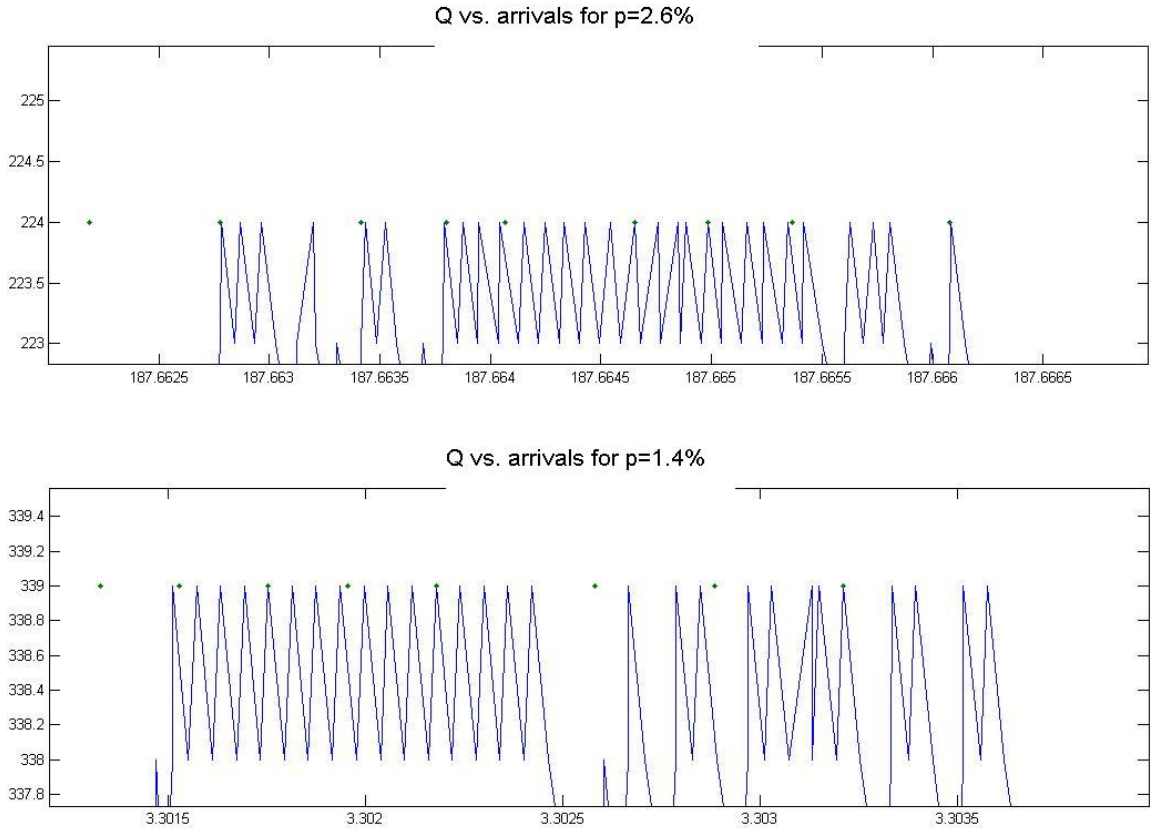

Figure 3.4: *The dynamical behavior of the Q during the peak period. The points correspond to the arrivals of the new packets. We mark a point evert 10 new arrivals. Two different cases of p are demonstrated here. The rates of the arrival and the fluctuations of Q have approximately the same form in both cases.*

state here an important theorem which will promote us to the future analysis of the *cwnd* distribution.

Theorem 4 *For congestion windows of size $w = n$, $p_e(n)/p = const$ independently of p.*

Proof 4 *We prove the theorem for some n. Denote by S_t the total number of windows with losses of size n, for some period t. In each window, 1 to n packets could be lost. Denote by β_t the number of times windows of size n were transmitted during that period. Then, according to the p_e definition, we have a per-packet packet loss event probability, which is equal for every packet in the window:*

$$p_e = \lim_{t\to\infty} \frac{S_t}{n * \beta_t} \tag{3.2.1}$$

Denote $E(l(n))$ the average number of lost packets $l(n)$ in a window of size n, given at least one packet in the window was lost. Since, using Assumption 14 the distribution of number of lost packets is constant and independent of n, its expected value is also constant and independent of n.

Denote as B the event that some arbitrary packet was lost. Denote as A_B the event that at least one packet was lost in the window which contained this arbitrary packet marked as B. *The general packet loss p, for some arbitrary packet observed in the network can be presented by Total Probability Law:*

$$p = Pr(B|A_B) * P(A_B) + Pr(B|A_B^C) * P(A_B^C) \tag{3.2.2}$$

Clearly, only the first term contributes to the p because $Pr(B|A_B^C) = 0$.

For the window of the size n, the probability of some packet in the window to be lost, then at least one packet was lost, is the ratio of the expected number of lost packets, given at least one packet was lost, to the total number of packets in the window, where Assumption 15 was used. This yields:

$$Pr(B|A_B) = E(l(n))/n. \tag{3.2.3}$$

The probability of the Packet loss event, according to the definition is:

$$P(A_B) = p_e(n) * n \tag{3.2.4}$$

Which is valid for any packet B. Finally, substituting the Equations 3.2.3 and 3.2.4 into Equation 3.2.2 we obtain:

$$p_e(n) = \frac{p}{E(l(n))} \tag{3.2.5}$$

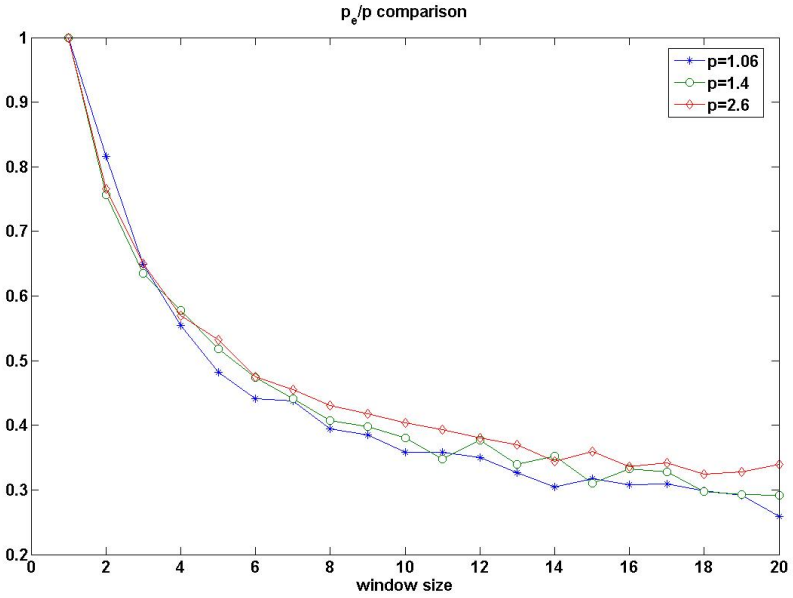

Figure 3.5: *Graph of pe/p*

The ratio presented here is derived from different scenarios of p, for w=1 to 20. The results were retrieved from the traces of NS2 simulations. For the larger windows it lacks the precision because they are too rare to happen. For example for p = 2% the window never exceeds 20 packets, so for higher window the results are quite occasional, as in the matter of fact they are irrelevant (the window cwnd reaches these values very rarely)

Since $E(l(n))$ is fixed, we obtain that $p_e(n)/p = const$, for any network, which proves the theorem. □

It is clear that once p is higher then p_e is higher as well, but the relation above stays constant. Note that we believe that this relation still holds even if we impose the scenario with larger B (order of rule-of-thumb [28]). Figure 3.5 gives the comparison for different scenarios and illustrates the Theorem 4. We may conclude, therefore, that the ratio of the packet loss event to the packet loss probability is constant for the dumbbell topology and we may use it for modeling the w for almost all possible scenarios.

So far we analyzed the correlation between the losses for the different $cwnd_i$ sizes. In order to accomplish the prerequisites we need, for defining the Markov Chain for the $cwnd^i$, we still lack the total probability of the loss for the window size $w = n$. We assumed in Assumption 15 for that purpose that for every window $w = n$, the loss probability for every packet is *independent*, however *it causes the packet loss event.* That is, the loss

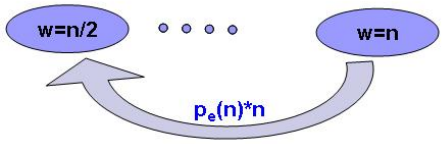

Figure 3.6: *Transition for the halving probability. At least on packet was lost. For the model with independent losses this probability is approximated as* $p * n$

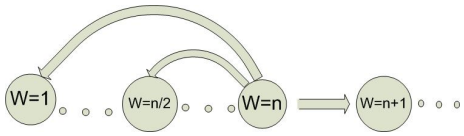

Figure 3.7: *Simple Markov chain chart*

may equally occur for any packet in the window, leading to the packet Loss Event as a consequence. The total packet loss, which leads to the window halving is equal to $pe(n) * n$ - as shown on Figure 3.6.
In fact our model is close to the model that utilizes independent losses, but instead of the expression $p * n$ we used $p_e(n) * n$ for the transaction which corresponds to halving the window size. We also utilize the Assumption 7, as we treat a unique source of all the losses in the entire topology, thus influencing the Markov chain of a *cwnd*.

The loss probability for every window rises with n. We neglect here the probability of timeout, and treat it separately. There is an alternative approach for utilizing the $p_e(n)$, which gives us a slightly more precise results, but it is more complex as it based on the details of TCP lossy dynamics. It is described in the designated subsection for the improvements of this model. Next we define the Markov Chain for $cwnd^i$.

3.3 Markov Chain for $cwnd^i$

We define the Markov chain, using 3 basic transitions, for the initial model, as shown in Figure 3.7. The transition from w to $w/2$ is the most common, and caused by losses which didn't lead to the slow start. Each of these transitions implies a packet loss event, and the total probability, as we found in the previous section, is equal to $pe(n) * n$. We rely here on the important quality of the Newreno dynamics. We assume that any number of losses in the window leads to a single window reduction and Newreno handles all the

losses within the same Fast Recovery and Fast Retransmit session. The transition to stage $w = 1$ is constant for small n and it is zero for the higher n. The reasonable solution for the TO is keeping a small table of constants for the TO for different p ranges for every state n. Assuming that the TO probability is constant leads to a small discrepancy in the mean of the final model, compared to the measure values. Finally we find the probability of increasing the window, by substraction of the other probabilities from 1.

3.4 Improvements and extensions to the model

We introduce in this subsection improvements for the implementation of the *cwnd* Markov Chain, we also discuss the Slow Start option. We present now a more complex way to find the transition of window halving using p_e, which relies more strongly on the nature of TCP Newreno. The window is halved once the 3rd sequential duplicate ack received. The Fast Retransmit is performed - transmitting the missing packet, and then Fast Recovery is started. Fast Recovery may inflate the window some more before it is actually halved. Some packets are sent when those two algorithms are active. The thorough explanations and examples for these two algorithms can be found in [24]. For simplicity and due to the Markovian limitations we assume that the window just immediately halved.

3.4.1 Halving Probability

We present here a new analysis of the Markov Chain transition probabilities. We would like to emphasize first the interpretation of the losses by the duplicate acks. In TCP packets are transmitted one by one, each packet for each successful ack, as long as the number of transmitted packets complies with the Slow Start or Congested Avoidance algorithms. The moment a predefined number of duplicated acks is received, the window is halved and the transmission process complies with the Fast Retransmit and Fast Recovery algorithms.

If the duplicated ack is received in the first half of the window, there is still space to accomplish the number of transmissions up to the half of the window, therefore additional packets are transmitted. One of the transmitted packets then, is the retransmission of the packet which caused a 1st duplicate ack (Fast Retransmit). Acks that are received on the second part of the window then, are disregarded, because we cannot transmit anything more anyway as a result of their arrival, because the window reached its maximum (it has been just halved).
On the other hand, if the loss happened in the second part of the window, we take into account all acks, that were accepted *till* the lost packet. In this case the new packet is

sent for each received ack and once the first duplicate ack received, the only additional transmitted packet will be the packet which is retransmitted during the Fast Retransmit , because the new window size (after the halving) wouldn't allow to perform any additional transmissions.

Therefore, we subdivide the computing of the halving probability to two cases:

- The lost packet event happened before we transmitted half of the current window. Denote the halving probability in this case as P_{h1}. In this case we do not care what happened with packets in the second part of the window. In case of additional losses, we just assume that there was no TO, and a normal process of Fast Retransmit and Fast Recovery handled the problem. If so, the window is halved (actually Fast Retransmit and Fast Recovery are started first). The w reduction probability in this case is:

  $$P_{h1} = p_e * (1 - p_e)^{w/2} * w/2 \tag{3.4.6}$$

  i.e. the event may happen in any place of this first half of the window, since anyway - the window is completed to half in this case. We assumed that the packet loss event affected only one packet, and that effectively p_e is independent for all packets in the window (in contrary to p) as we discussed in the p_e derivation. The expression $p_e*(1-p_e)^{w/2}$ stands for the probability of a single loss in the first half of the window, i.e. one packet was lost with probability p_e and $w/2$ packets were transmitted correctly. Since p_e reflects effectively independent losses, and it is reasonably small, the probability of two losses in this half (and in the entire window as well) is negligible. The location of the lost packet may be anywhere in the first $w/2$ packets and that is why we multiply by $w/2$.

- The lost packet event happened after the current window transmitted its half or more. Denote the halving probability in this case as P_{h2}. In this case the retransmission happens in the next window, and the new packets are transmitted during the Fast Recovery , only if cwnd allows it. We conclude that in this case the window reduction probability is:

  $$P_{h2} = \sum_{i=w/2+1}^{w-1} p_e * (1 - p_e)^i \tag{3.4.7}$$

  Explanation: we sum all cases in which more than half of the packets were acked first, and then the lost event occurred. The location of the duplicate ack was the last, while acks that were accepted before were successful. That is why we do not take into account here the location of the duplicate ack since it was the last one. The last packet in the window is not counted as well because it refers to the Fast Retransmit , which belongs already to the next window.

Finally, as the two cases are independent, and in a case of loss either one of them can happen the halving probability will be:

$$P_h = P_{h1} + P_{h2}. \tag{3.4.8}$$

3.4.2 Slow Start

Next step to improve the model is considering the Slow Start algorithm. We apply this improvement to the first model with "first order" steady states already computed. Slow Start happens only after timeout. So we count the probability for states $w_i = 2, 4, 8, 16, 32$. The maximum window size is $w_{max} = 64$, typically. We call those states *states of interests*, and for them we compute the Slow Start probability. The condition that the next increment of the window will be Slow Start and not the Congestion Avoidance is, that the last fall obeys both of the following conditions:

1. The window was reduced to 1 as a result of TO (timeout).
2. The last window before TO was accepted was at least twice the current window (in order to make the ssthresh to allow the Slow Start increment up to this window size)

Both conditions are dictated by TCP dynamics. We compute the Slow Start probability by observing what happens after the last loss. Denote by n the size of the state of interest. We first find the probability that the last loss will allow the window to be below n. We will distinguish three loss cases:

- A window halving loss followed by Congestion Avoidance.
- A timeout loss, starting from a window size of less than 2n, that would therefore not go through the transition $n \to 2n$ in the Slow Start.
- A timeout loss, starting from a window size of above 2n, that would therefore go through the transition $n \to 2n$ in the Slow Start.

Denote as π the steady state probability which we found first, without Slow Start, CA_n is the transition probability for incrementing the window to $n + 1$ from state n, H_n the probability of window halving, and TO_n will be the probability of TO in state n. The probability that the fall caused a timeout (from any other state) is then:

$$P(Timeout) = \sum P(Timeout|n) * \pi_n = \sum_n \pi_n * TO_n \tag{3.4.9}$$

The probability that the next transition will be an increment by 1(CA), happens when the loss resulted from window halving, is given as follows:

$$P(CA_n) = \sum P(CA_n|m) * \pi_m = \sum_{m=2}^{n*2+1} H_m * \pi_m \tag{3.4.10}$$

Clearly, states that are above $2n+1$ are irrelevant (otherwise the fall will result in states that are higher then the one we are interested in). We are interested in the probability then TO happened in states that are at least $2 * n$. For states that were lower, the next ssthresh value wouldn't allow to go high to the state of interest by Slow Start, because the Congestion Avoidance will start before reaching the state of interest, due to the ssthresh limit.

$$P(Timeout_n) = \sum P(Timeout_n|m) * \pi_m = \sum_{m \geq 2*n} \pi_m * TO_m \tag{3.4.11}$$

Clearly, holds $P(Timeout_n) < P(Timeout)$, because in the first argument of the inequality we count only a part of all the timeouts. The total probability that the last loss will cause to reach the state of interest(either by Slow Start or by Congestion Avoidance) is $P(CA_n) + P(Timeout)$. The probability to get to the state of interest by Slow Start, i.e of reaching the window of size n by Slow Start, is therefore:

$$P(Sl_n) = P(Timeout_n)/(P(CA_n) + P(Timeout)) \tag{3.4.12}$$

The probability for increasing the window in the state of interest is the same, and we can split it now to two different probabilities: pass the state of interest by Slow Start, or by Congestion Avoidance. If so, we change the steady state probability for the relevant states, by reducing the CA_n by weighting it by $(1 - P(SL_n))$, and adding the new transition to the $n * 2$ state, with probability $CA_n * P(Sl_n)$.

We present the comparison of the model, with the model including the extensions (second approach for the window decrease) and the simulated statistics in Chapter 5. Very close pdf form and good results for the mean and variance are achieved, for low, high and medium packet loss.

3.4.3 Convergence of the Markov Chain

We would like to prove next the convergence of the Markov Chain (MC) which we constructed to the steady states.

Lemma 1 *All states of the MC are positive recurrent*

Proof of Lemma 1 *To prove recurrency we need to show that:*

$$\sum_{k=1}^{\infty} P_{ii}^{k} = \infty \tag{3.4.13}$$

where P_{ii}^{k} is the probability to get back to state i in k steps. Denote the relation $i \leftrightarrow j$ for two states i and j as communicating *states. It is trivial that $1 \leftrightarrow j$ for all j 1 to w_{max}, where w_{max} is the maximum allowed congestion window, since it is possible to get to each state from state $cwnd = 1$. Since it holds*

$$\sum_{k=1}^{\infty} P_{11}^{k} = \infty \tag{3.4.14}$$

state $cwnd = 1$ is recurrent. The last equation is due to the fact that for all k, $P_{11}^{k} > \varepsilon$, because it is assumed that in any state at time $k-1$ there is a positive probability ε to get a timeout, therefore, considering all states is enough to show that the Equation 3.4.14 holds. Since if $i \leftrightarrow j$ and i is recurrent, then j is recurrent as well - all states are recurrent. The expected value of time of getting back to each state is finite therefore all the states are positive recurrent. □

Lemma 2 *The MC of cwnd is non-periodic.*

Proof of Lemma 2 *State i is be periodic with some period d if $P_{ii}^{n} = 0$ for any n which satisfies n* modulo *$d \neq 0$. Since there is no such d for any state - all states are non-periodic.*
□

Corollary 1 *Since the MC is positive recurrent and non-periodic it is ergodic*

Lemma 3 *The MC of cwnd is irreducible.*

Proof of Lemma 3 *For all couples of states i and j exists $k \geq 0$ and $m \geq 0$ such that $P_{ij}^{k} > 0$ and $P_{ji}^{m} > 0$. The statement above holds because there is a positive probability for TO , and state $cwnd = 1$ communicates with every state.* □

Theorem 5 *MC of cwnd converges to its steady state, i.e. the limit*

$$lim_{n\to\infty} P_{ij}^{k} = \pi_j \tag{3.4.15}$$

exists and is independent on state i.

Proof 5 *The proof stems from Lema 3 and Corollary 1*
□

Chapter 4

Towards the full model of the traffic

We finished the closed loop of probability models by deriving the packet loss rate, proving that the total arrival rate to the queue is Gaussian. We would like to present in this chapter the analysis of the traffic on the Lines of the dumbbell topology, as was demonstrated in Figure 2.2. We first find the model of L_1. Then, we present the distribution of the size of the burst which travels the lines after the Q. We use this distribution to find the traffic on L_2, L_4,L_5,L_6, and finish with a model for W - the sum of all cwnd.

4.1 Model of L_1

The L_1 is the only part of the network which is *before the bottleneck*, i.e. no packets were lost yet. Knowing the distribution of l_1^i we can easily find the distribution of L_1. Remember, that the $cwnd^i$ of different flows are independent (Assumption 8), therefore we can assume the same about the l_1^i. We derive L_1 by following theorem:

Theorem 6 $L_1 \sim Norm(\Sigma_i E(l_1^i), \Sigma_i var(l_1^i))$

Proof 6 *l_1^i are distributed according to the same function, but the parameters are different (Theorem 1). Therefore we have to prove a Lindeberg condition in order to use the Central Limit theorem. The proof is identical to the proof of Theorem 3. We just will show here the Lindeberg condition in this case:*

Denote as η_i the expected value of the number of packets on the client forward access

link $E(l_1^i)$*, and denote as* σ_i *its standard deviation. We define the following sum :* $s_n^2 = \sum_{i=1}^{n} \sigma_i^2$*. Then, for every* $\varepsilon > 0$ *holds:*

$$\lim_{n\to\infty} \sum_{i=1}^{n} E(\frac{(l_1^i - \eta_i)^2}{s_n^2} : |l_1^i - \eta_i| > \varepsilon \cdot s_n) = 0 \tag{4.1.1}$$

the proof continues identically to the proof of the Theorem 3.
□

We bring the comparison of the modeled L_1 to the measured one in the Chapter 5.

4.2 Distribution of bcwnd

We found cwnd in the previous chapter and we showed that the packets are traveling on the links in a bursty manner. The packets are starting their way on the dumbbell topology on L_1, when the burst size is equal to cwnd - the size of the congestion window. When this burst passes the queue, some packets might be lost, according to the packet loss distribution. We call *bcwnd* the distribution of the size of the burst *after passing the queue* in the buffer. In this section we find the pdf of *bcwnd*, and then we will be able to use it to find the distributions of the traffic on the Lines in the next section.
The best approach to find the distribution of *bcwnd* is using the distribution of $l(n)$ which is equal to $p/pe(n)$, according to the Theorem 4. The initial size of *bcwnd*, before entering the queue is equal to *cwnd*. We construct a transformation that shows all possible transitions, from *cwnd* to all possible sizes of *bcwnd*. The diagram of this transition is illustrated in Figure 4.1. For each *cwnd* of size n the number of packets that can be dropped ranges from 0 to n, according to the distribution of $l(n)$. We deduct the probability of $P_{loss} = p_e(n) * n$ from the state $bcwnd = n$ and distribute it among the other states of $bcwnd = i, i < n$. This transformation is activated *only once* for each burst of size *cwnd* when the burst of size cwnd passes the buffer, once in a rtt. Therefore each time only one transition is possible, according the distribution shown in the illustration. The update of the pdf of *bcwnd*, according to the presented transformation is done starting from $n = 1$ till $n = 64$. A short description of the update algorithm is as follows:

1. $P_{loss}(n) = p_e(n) * n$ (TO is small and omitted)
2. For each n from 1 to 64 update:
3. $Pr(bcwnd = n) = Pr(bcwnd = n) - Pr(cwnd = n) * P_{loss}(n)$

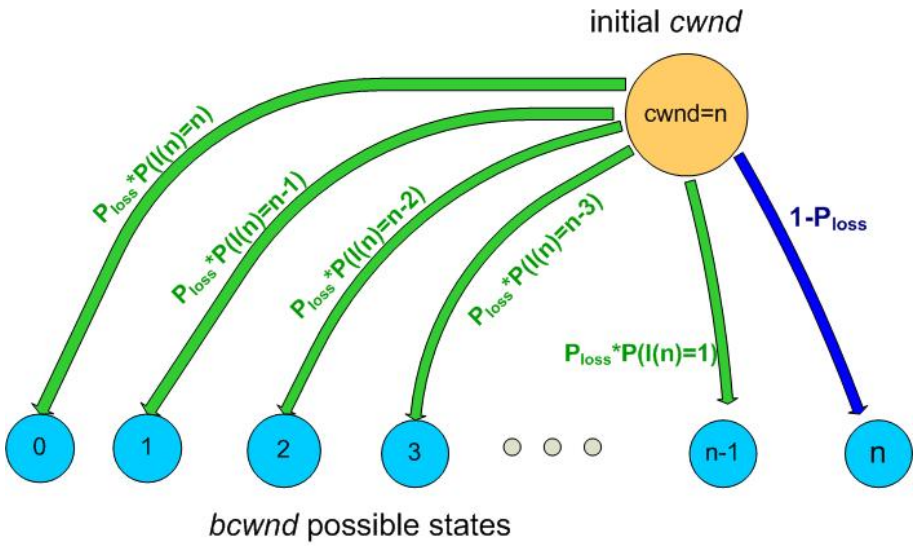

Figure 4.1: *Transition from cwnd to bcwnd. The rightmost transition stands for the no-loss case. The leftmost transition is then bcwnd is zero, i.e. all packets in the window were lost. The total distribution of bcwnd is achieved then this chain is applied for all cwnd (of all possible n)*

4. For each i, i=0 to n-1 update: $Pr(bcwnd = i) = Pr(cwnd = i) + Pr(cwnd = n) * P_{loss}(n) * Pr(l(n) = n - i)$

Alternatively we may represent the transformation from $cwnd \rightarrow bcwnd$ by the following formula:
(4.2.2)

$$Pr(bcwnd = n) = Pr(cwnd = n) * (1 - P_{loss}(n)) + \sum_{k=n+1}^{64} Pr(cwnd = k) * P_{loss}(k) * l(k - n)$$

The difference in the two presentations is the complexity of implementation, as the algorithm is easier to implement. That is, if we represent $l(n)$ for all n as a matrix we should treat diagonals instead of rows or columns.

The expected value and the variance of bcwnd is smaller then that of cwnd as it shown on Figure 4.2

We can use now this pdf for the probability models of the traffic on the Lines.

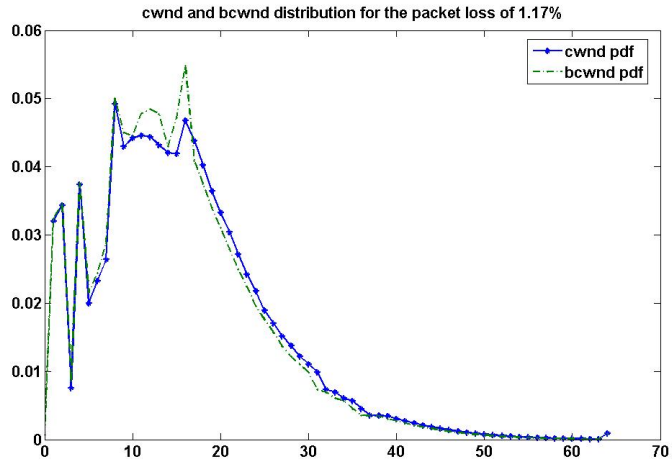

Figure 4.2: *Distribution of bcwnd*

4.3 Distribution of L_2,L_3,L_4,L_5,L_6

In this section we derive distributions of L_2,L_3,L_4,L_5,L_6, as previously shown in Figure 2.2. We start with L_3 because its traffic has special property. In fact, the only bottleneck is before the L_3. The packets injected to the L_3 are emitted by the queue. The capacity of this line is limited, because the rate of the emission C is limited as well. We approximate that the occupancy of L_3 therefore is most of the time at its maximum and therefore *constant*.

L_5 is the next line after L_3 for the packets to travel. Since the server forward access links are not equal but distributed uniformly, the number of packets on each line l_5^i is independent. The fact that all the packets leave the same emission point - inserts some dependency. Still, assuming that the l_5^i are independent we can show that L_5 can be approximated as Gaussian as well, using the same method as we did for L_1. In the lines L_6 and the following L_4, which include only acks, the dependency is weaker , because the packets to these lines are injected from the different sources. We find the distribution of l_2^i, l_4^i ,l_5^i, l_6^i exactly by the same method which we used to find l_1^i, with the only difference that we use *bcwnd*. Next, we utilize Lindeberg Cental Limit theorem again, to find the Gaussian distributions of L_2,L_4,L_5,L_6, according to l_2^i, l_4^i ,l_5^i, l_6^i respectively. We compare these models with an alternative approach, and discuss the precision in Chapter 5.

4.4 Distribution of W

Denote as well the common window distribution: $W = \sum_i cwnd^i$, where $i = 1...N$, N is the number of clients. We may express the total number of packets circulating the topology as follows:

$$T = L_1 + L_2 + L_3 + L_4 + L_5 + L_6 + Q \tag{4.4.3}$$

For a small packet loss we may neglect the percentage of the lost packets, stating approximately $T = W$. Utilizing the statistical independence of $cwnd^i$ we can model the W. Denote the $w^i(t)$ as a distribution of process $cwnd^i(t)$ of every flow.

Theorem 7 *The sum W of cwnd of all flows has a Gaussian distribution.*

$W \sim Norm(\Sigma_i E(w^i), \Sigma_i var(w^i))$

Proof 7 *Since all the flows are modeled as independent (Assumption 8), and equally distributed (Assumption 9, from the Central Limit Theorem we deduce:*

$E(W) = \sum_n E(w^i(t))$, $var(W) = \sum_n var(w^i(t))$ *The sum of i.i.d. random variables converges to a Gaussian distribution.* □

Therefore, once we found the distribution of $w^i(t)$, we have complete statistics for W, which is, according to the Central Limit Law, Gaussian distributed. We stated earlier that for low packet loss approximately $W = T$. We can refine the approximation by assuming that W is a delayed equivalent of T and the difference is the lost packets, which are not considered in W. We thus have: $W \approx \frac{TOTAL}{1-p}$.

Chapter 5

Simulation Results

In this chapter we present simulation results of the entire model and explain the differences between various cases. We performed NS2 simulations for several hundreds flows (500 to 2000) with dumbbell topology. We randomized the propagation times of access links, thus making the tp_1^i,tp_2^i,tp_5^i,tp_6^i, to be uniformly distributed (Assumption 5) within different ranges ($20 - 100$ msec. for the lower bound and $50 - 500$ msec. for the higher bound). The propagation time of the bottleneck link tp_3, tp_4, was 20 msec. We implemented also a small percentage (about 5%) of short TCP flows, to create some noise, but we did our measurements only with long-provisioned TCP flows. We sampled the window size of all flows, and also measured the traffic distribution on the Lines. Sampled vectors give us a histograms, which lead to the normalized pdf. We present first the results of the *cwnd* model. Next we discuss the transmission rate results. Finally we introduce the graphs for L_1 and show the Gaussian functions for all the parts of the network. An additional paragraph on the influences of changing the parameters of the network and special cases concludes this section.

5.1 CWND Model Simulation Results

We verified with the simulated results the precision of the model of W by applying both approaches for the halving probability which we presented. We bring here the CDF and PDF for 3 different cases of p. Figures 5.1, 5.2, 5.3 represent the PDF for low, medium and high packet losses, while Figures 5.4, 5.5, 5.6 represent the CDF for the same cases. We can see that for all three packet losses the model is very close both in CDF and PDF. The alternative approach for the halving calculation improves slightly for the medium and

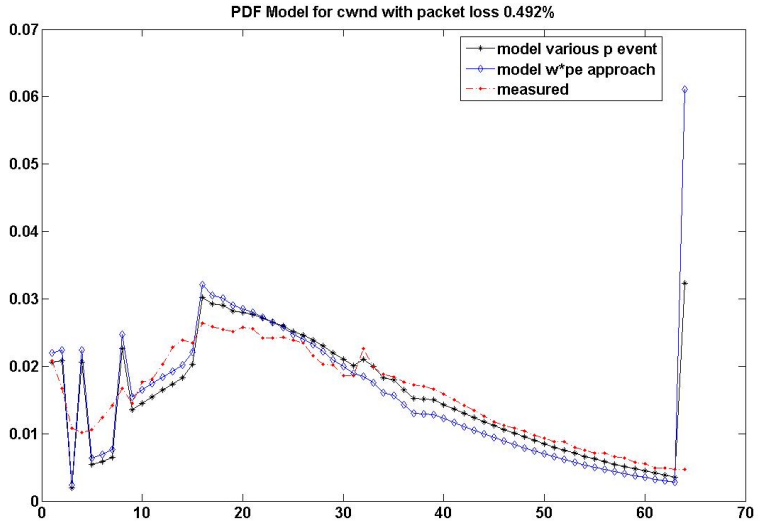

Figure 5.1: *Probability Distribution Function, low packet loss case.*

high packet loss. We conclude that the model is correct for the entire range of interest of packet losses.

5.2 Gaussian Model of W and Access Links

We present the comparison of our probability models of W, and Lines, to the measured results. We used a model of $cwnd^i$ and l_1^i to produce the Gaussian models. First we find the model for *cwnd* using a given p. Second we find the distributions of all l_1^i using Equation 2.1.1. Finally we use Theorem 3 to find the L_1 parameters using the Central Limit Law. Figure 5.7 demonstrates the result. In spite of the models not being 100% precise the results for L_1 are still close enough. The most influencing result on the entire model is the mean of *cwnd*. The skew of order of 1 packet can cause the shift of about 5% of the mean of L_1. Therefore, as can be observed, in this example the model of *cwnd* yielded very precise results.

Next we show the results of the model of L_2, which we computed using *bcwnd*. Typically, as we mentioned in the introduction to this section, tp_k^i might be distributed differently for all $k, k = 1...6$. We compare our model based on *bcwnd* with an alternative approach as follows: We take the already produced model of L_1, we multiply it by factor $1 - p$ and then scale it by the ratio of means of propagation times, which yields an equation as

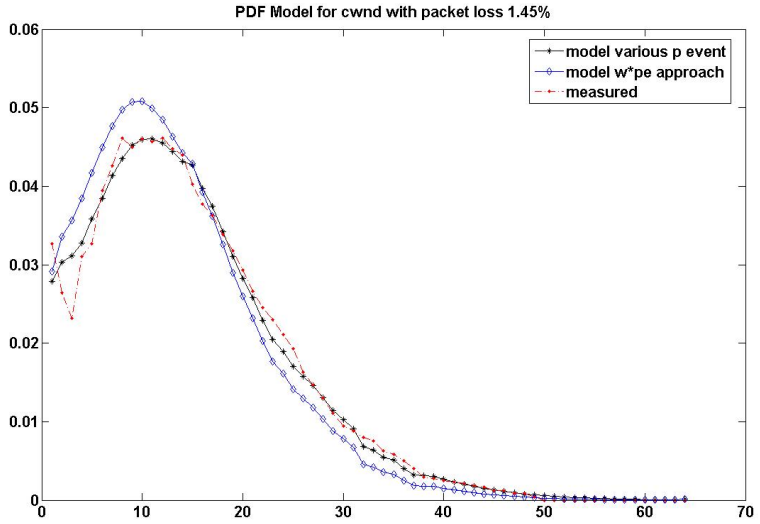

Figure 5.2: *Probability Distribution Function, medium packet loss case.*

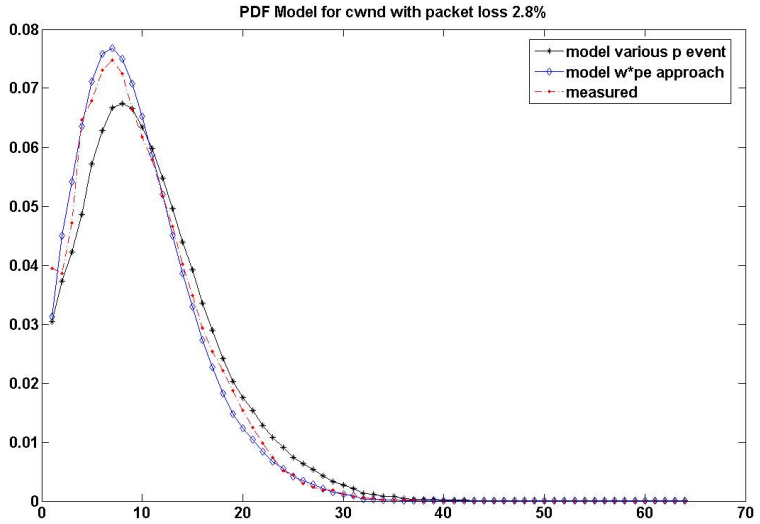

Figure 5.3: *Probability Distribution Function, high packet loss case.*

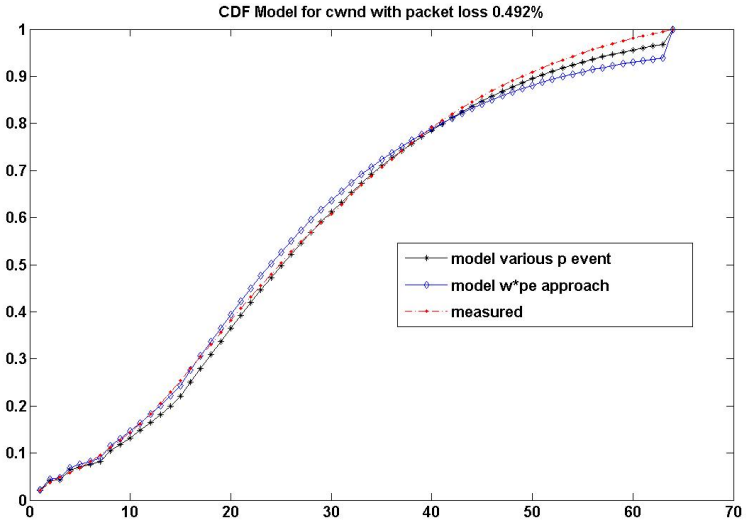

Figure 5.4: *Cumulative Distribution Function, low packet loss case.*

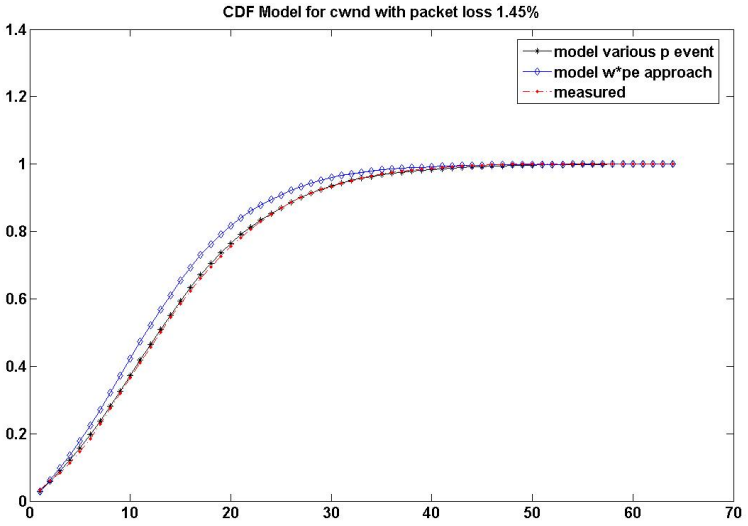

Figure 5.5: *Cumulative Distribution Function, medium packet loss case.*

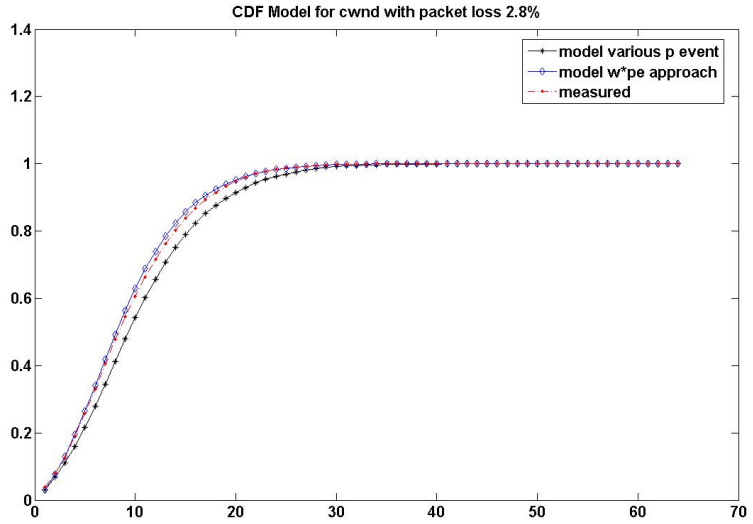

Figure 5.6: *Cumulative Distribution Function, high packet loss case.*

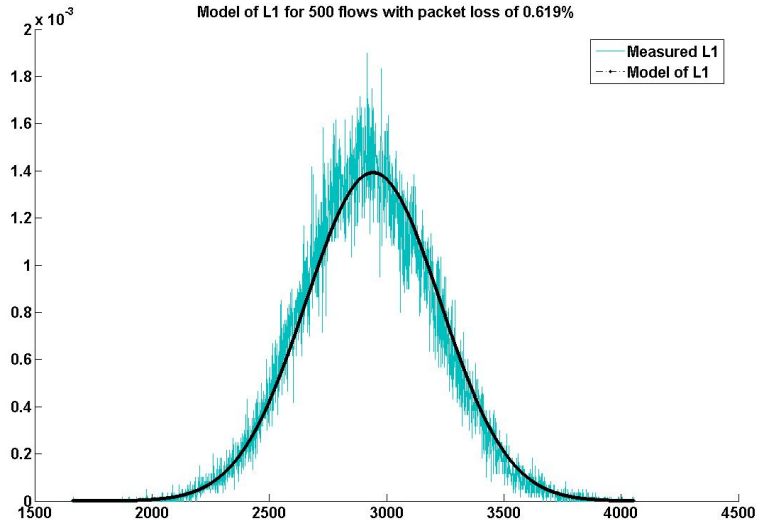

Figure 5.7: *Model of* L_1 *for* 500 *flows, low packet loss*

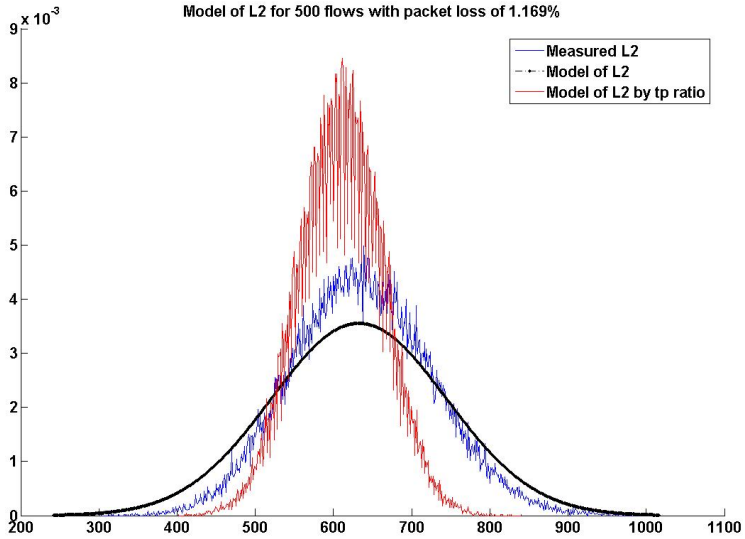

Figure 5.8: *Model of L_2 vs. alternative approach.*

follows:

$$L_2(t) = L_1(t) * (1-p) * \frac{mean(tp_2^i)}{mean(tp_1^i)} \tag{5.2.1}$$

Figure 5.8 demonstrates the comparison. Comparing this alternative with our model we see that it lacks the precision both in expected value and in variance. The main reason for that is that scaling by the propagation times mean is not reliable and depends on the distribution of propagation times.

It is worth mentioning that our model for L_2 is also less precise then the model of L_1, for instance. The reason for that is that different l_2^i have some small correlation. Unlike the traffic of L_1, the traffic of another links is influenced by the common factors - queue and L_3 as a common Line, which are non Gaussian and insert some correlation. We assumed a zero correlation which is correct for L_1, but is an approximation for the other lines. We discuss further the influence of this correlation in this chapter as well.

The example of the Gaussian distributions of Lines and W is demonstrated on Figure 5.9, where we can see 5 Lines that comply with the Gaussian distribution, together with the distribution of W.

Next we discuss the result for the model of W in detail. We use Theorem 7 to find the distribution of W. Figure 5.10 demonstrates the result, when the transmission rate

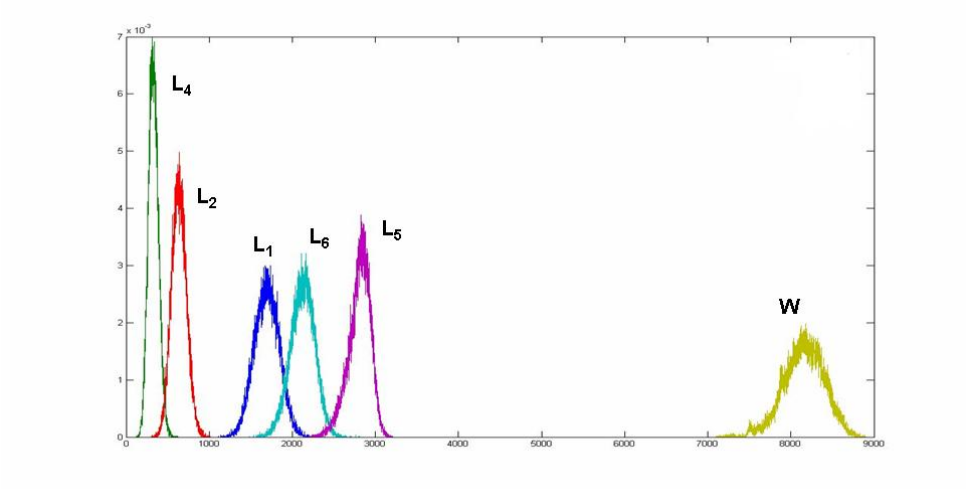

Figure 5.9: *Gaussian distributions of the packets on the links*

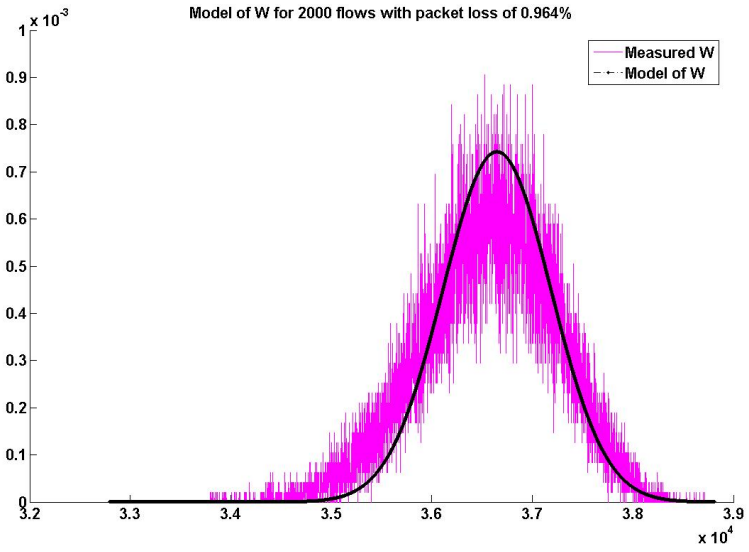

Figure 5.10: *Model of W for* 2000 *flows, low packet loss, not significant B*

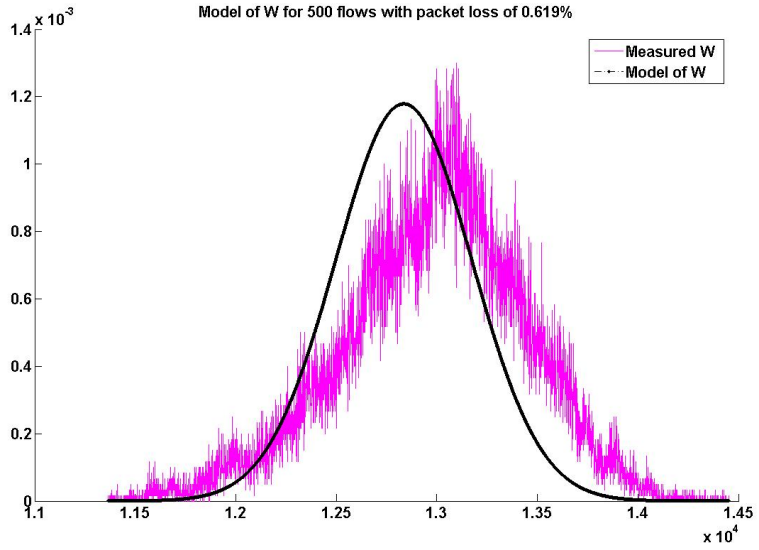

Figure 5.11: *Model of W for* 500 *flows, low packet loss, significant B*

is high, while B is quite high as well (approximately of a size that complies with the Stanford Model). The model looks quite precise in spite of the quite large B. The reason that B is not significant is that due to the high rate, the number of packets in the queue is comparatively negligible. Next we present a case with 500 flows with much lower total rate, but a more significant B. Figure 5.11 demonstrates the result. The influence of a non Gaussian factor is quite easily observed. The model looks now a little bit let precise because of the non zero covariance between the $cwnd^i$. The difference is still very small (less than 5% in mean and looks large because of the wide scale). The covariance between two arbitrary flows is low (the covariance coefficient is less than 0.01, but the accumulative influence of the total covariance is clearly seen, so the variance of the measured W is higher for 500 flows, than the modeled. This is in contrast to the model of L_1 of the same case (Figure 5.7). The reason for such a difference in precision is that L_1 is not directly affected by the queue, while W contains it.

5.2.1 Gaussian tests

We checked the precision of the measures of the number of packets on all the line with the QQ-plot method [13] and also with the Lilliefors test [17]. The QQ-plot results for different Lines presented on Figure 5.12. Our estimations about the compliance of different Lines with the Gaussian distribution were confirmed by these results.

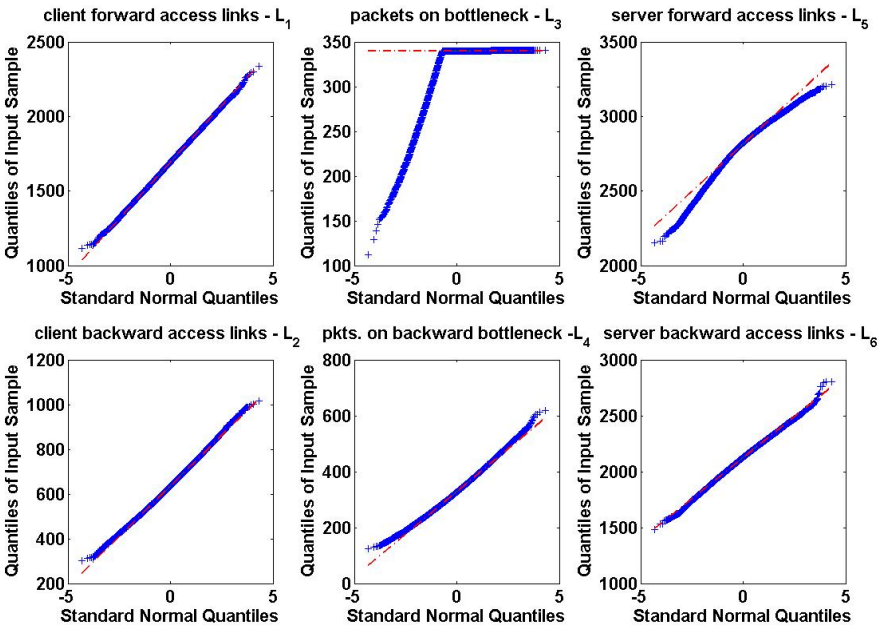

Figure 5.12: *Gaussian tests with QQ-plot MATLAB tests. The test was performed for all 6 lines. The only clearly non Gaussian line is Line 3 - the bottleneck right after the buffer. Some of its influence is seen as well on the next line - L_5, which has some small deviations in the 5th quintile*

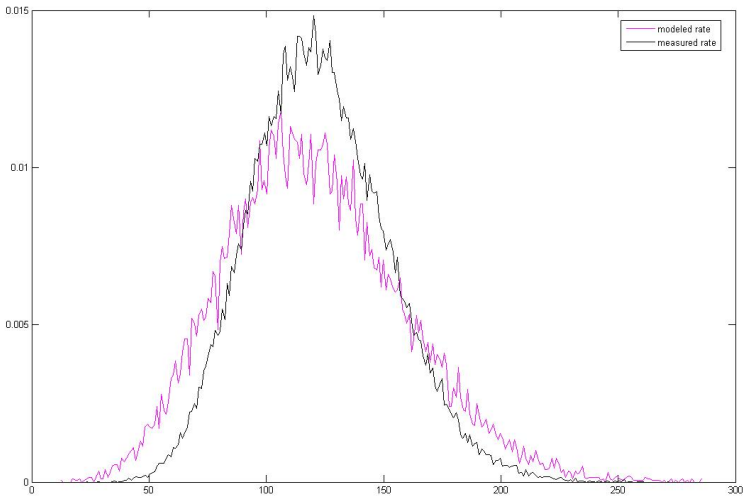

Figure 5.13: *Example of Arrival Rate pdf, for the* $p = 2.3\%$ *achieved with 500 flows. We can see that our model gives a slightly higher variance, due to the burstiness assumption.*

5.3 Rate Model

The modeled total rate yields a result that is quite close to the measured, as we saw in Figure 2.6. We can also see that the more flows we have, the more the discrepancy in variance in our model tends to vanish, compared to Figure 5.13. The effect of the burstiness is less visible for the larger number of flows.

5.4 Changing the Parameters of the Network and Special Cases

The network topology is actually defined by set of fixed and small number of parameters that we already presented: C, B , $\{rtt_i\}$. We will now discuss the effects of changing these parameters.

5.4.1 Flows with Different Link Service Rates

Among the special cases we may consider adding a capacities on different forward client access links $\{C_i\}$, limiting the link capacity of the flows as well. Clearly, this limitation

relaxes a burden off the bottleneck. For very low $\{C_i\}$, then C is greater then $\sum C_i$, the packet loss on the bottleneck is always zero and the Queue is empty. When the C_i are limited but not that low, the packet loss drops down, but the nature of the probability function stays the same.
The difficulty which might rise here is the modeling of $cwnd^i$. The window maximum may now be smaller then 64 packets, or any other maximum predefined during the TCP connection, due to the limited line which won't be able to carry 64 packets at once. We assume that in this case the window maximum will be redefined, and in case we have flows with different maxima the *cwnd* distribution must be calculated separately. There is no problem in finding the distribution of W after that - instead of general Central Limit case the Lindeberg condition must be proven again. The l_1^i, L_1 and the rate models will stay with no change, and all the functions remain Gaussian. We confirmed this with NS2 results.

5.4.2 Topology with Single Server

We researched the case when all the flows connect to a single server, all tp_5^i are equal, and all tp_6^i are equal as well. The traffic which exits L_3 is not split but travels till the common server. The traffic on the lines L_3, L_5,L_6 stays in this case non Gaussian and has the same distribution as on L_3, scaled by the ratio of the propagation times. The Gaussian distribution stays on L_1 and L_2, and W is Gaussian as well.

5.4.3 Same Propagation Times

We can simplify the dumbbell topology by assuming all tp_k^i are equal for all $k, k = 1...6$. In this case only L_1 and W preserve the Gaussian quality. Other Lines have a distribution form which is identical to L_3. The reason for that is that distribution of l_2^i are identical now. l_1^i are transmitted asynchronously from different sources and stay i.i.d. In case we synchronize the flows by starting them all exactly in the same moment - L_1 will lose its Gaussian property as well, and so does W.
It is worth mentioning that all these cases are strictly theoretical, and are unlikely to happen in a real network

Buffer	Formula	p (%)	Avg cwnd
154	C*RTT/sqrt(N)/17.5	0.8	21
772	C*RTT/sqrt(N)/3.5	0.664	24.5
3862	C*RTT/sqrt(N)/0.7	0.45	28.5

Table 5.1: *Buffer Size Influence for a Low p*

Buffer	Formula	p (%)	Avg cwnd
154	C*RTT/sqrt(N)/8	3.26	8
450	C*RTT/sqrt(N)/2	2.98	8.8
1802	C*RTT/sqrt(N)/0.5	2.01	11.5

Table 5.2: *Buffer Size Influence for a High p*

5.4.4 Buffer Size Impact

We promoted the usage of small buffers - we would like to explore the justification of this approach. We present in Table 5.1 and in Table 5.2 comparisons of performance for 3 categories of buffers - large (rule of thumb), medium (Stanford model) and small (Stanford model divided by constant). We can see that the improvement in the performance with larger buffers is not dramatic, even if we do not consider the queuing delay. We compare two parameters for the performance - average *cwnd* and the packet loss. The improvement in both parameters was insignificant, comparatively to the increase in the buffer size. As long as queuing delay is negligible (Assumption 11), the throughput will change linearly with the average *cwnd*, provided that rtt^i are similarly distributed. Therefore, improvement in throughput will be close to the improvement in average *cwnd*.

In order to track further the influence of the buffer particularly on the packet loss we changed the buffer size for the same topology. The results are presented in Table 5.3. Graphical presentation is given in Figure 5.14. We can see that significant improvement happens for the very small buffers. When the buffers are very large, they simply absorb a major part of the total needed requirement, bringing the average *cwnd* quite close to $w_{max} = 64$. However, comparing the average *cwnd*, which is proportional to the throughput (neglecting the queueing delay), we see very slow improvement, as it shown on Figure 5.15.

B (packets)	7724	3862	1931	772	386	154	67
p (%)	0.00255	0.0045	0.00559	0.00664	0.00755	0.008	0.01025
avg. cwnd	38	28	26.5	24	22.5	22	19

Table 5.3: *packet loss as a function of the buffer size*

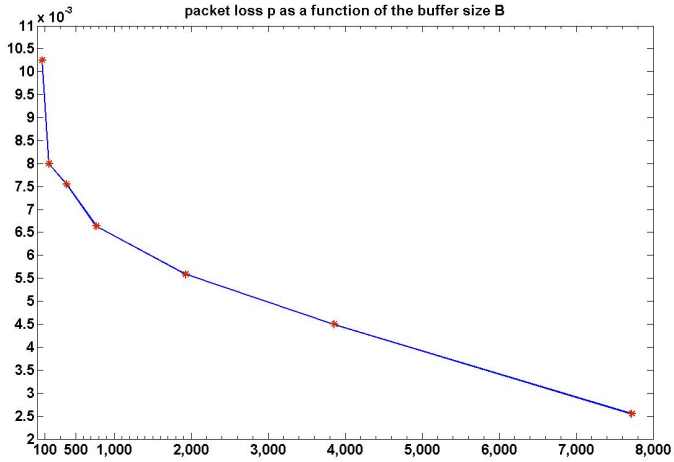

Figure 5.14: *packet loss as a function of B*

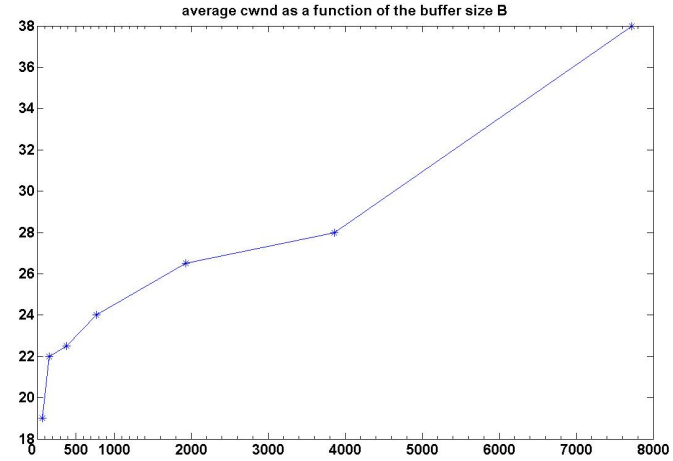

Figure 5.15: *average cwnd as a function of B*

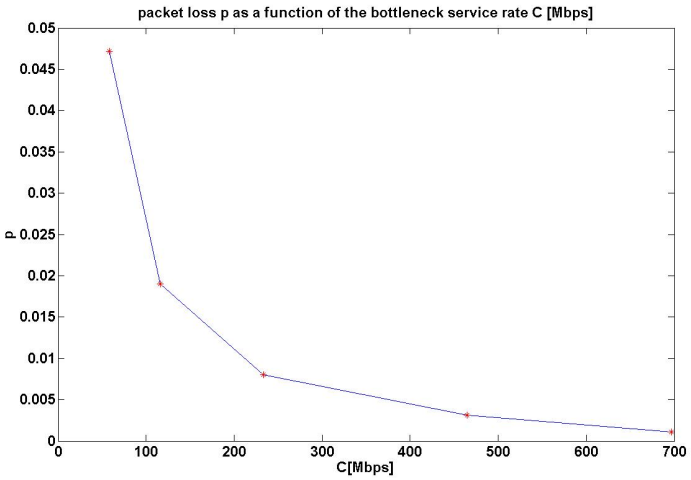

Figure 5.16: *packet loss as a function of C*

5.4.5 Bottleneck Service Rate Impact

An additional interesting influence we investigated was that of the bottleneck link service rate C. Unlike the impact of increasing the buffer, which is not straightforward and causes very slow improvement in the packet loss p, the average *cwnd* and consequently the throughput of each flow, the influence of C is straightforward and fast. Increasing C leads quite quickly p to zero and cwnd to $w_{max} = 64$ packets. The decrease in p nearly has an exponential form. The improvement in *cwnd* would have an exponential form as well, once approaching to w_{max}. The results are demonstrated on Figure 5.16 and Figure 5.17.

Once the service rate is more than the maximum total arrival rate R, no packet loss will happen in the network and *cwnd* will be constantly equal to its maximum value.

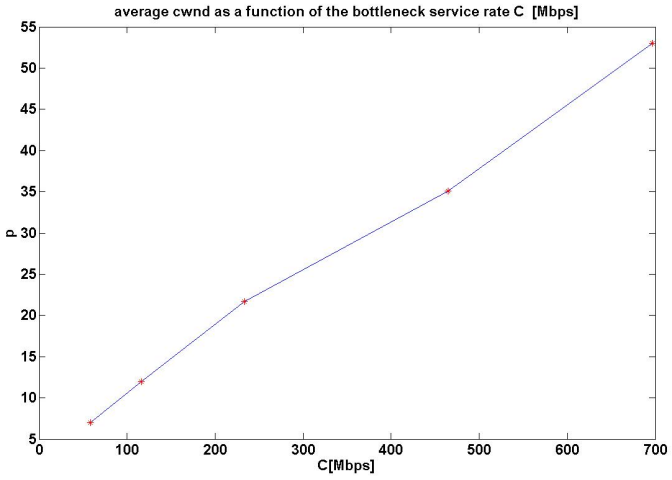

Figure 5.17: *average cwnd as a function of C*

Chapter 6

Conclusion

In this chapter we bring the summary and useful implications of this work.
We constructed a closed loop of models, which statistically describes the dumbbell topology. We started with a model for the traffic on a single access line and proceeded to the model of the rates on these lines. We found then the model of the total arrival rate. Using this rate distribution, we found by G/D/1/K analysis the distribution of Q. The distribution of Q led us to the packet loss p. We found the packet loss numerically by solving the fixed point equation $p = f(p)$ where $f(p)$ represents the closed loop of models.

We completed the closed loop, in fact, in chapter 3, which was dedicated for the model of *cwnd*. We explored the nature of the correlated loses and found distribution of *cwnd* by simple Markov Chain, using two different approaches.

Finally, we accomplished the statistical description of the dumbbell topology by finding the distribution of the traffic on all Lines, proving that all but the bottleneck line comply to a Gaussian distribution.

The closed loop and the topology description we found can serve two objectives.
First, we obtained a network planning capability. We can fully describe the dumbbell topology by knowing the buffer size, bottleneck link capacity and the set of rtt of different TCP sources. We can change at will the link capacity or the buffer size in order to achieve a certain demand for the packet loss. We can reduce a buffer size in order to maintain some level of a packet loss as well. In summary - we defined a rule for the tradeoff - *buffer size vs. bottleneck link capacity vs. packet loss*. This rule might in the closest future serve the router planners.
Second, the complete description of the traffic gives as a possibility to track the problems in the existing networks. The loop of models gives an insight into the network performance.

We can compare different networks by their average *cwnd* and by the mean and variance of the traffic on their lines as well.

Bibliography

[1] A. Abouzeid, S. Roy, and M. Azizoglu, *Stochastic Modeling of TCP over Lossy Links*, Proceedings of INFOCOM 2000.

[2] E. Altman K. E. Avrachenkov A. A. Kherani B. J. Prabhu, *Performance Analysis and Stochastic Stability of Congestion Control Protocols* ,INRIA Report No. RR-5262, Sophia-Antipolis, France, July 2004

[3] E Altman, K Avrachenkov, C Barakat, TCP in presence of bursty losses. ACM SIGMETRICS, Santa Clara, CA, Jun. 2000

[4] Avrachenkov, Altman, Barakat, *The effect of router buffer size on the TCP performance*, Proceedings of LONIIS workshop on Telecommunication Networks and Teletraffic Theory, St.Petersburg, Russia, pp.116-121, January 2002.

[5] Guido Appenzeller, Isaac Keslassy and Nick McKeown, *Sizing Router Buffers*, ACM SIGCOMM '04, Portland, Oregon, September 2004. Also in Computer Communication Review, Vol. 34, No. 4, pp. 281-292, October 2004.

[6] Guido Appenzeller, Nick McKeown,Joel Sommers, Paul Barford, *Recent Results on Sizing Router Buffers* Proceedings of the Network Systems Design Conference, San Jose, CA, October 2004

[7] Olga Bogoiavlenskaia, Markku Kojo, Matt Mutka, Timo Alanko, *Analytical Markovian model of TCP, Congestion Avoidance Algorithm Performance*, Series of Publications C. Report C-2002-13. Department of Computer Science, Univiersity of Helsinki, Finland.

[8] Amogh Dhamdhere, Hao Jiang, Constantinos Dovrolis *Buffer Sizing for Congested Internet Links*, Proceedings of IEEE Infocom, Miami FL, March 2005

[9] Vincent Dumas, Fabrice Guillemin, Philippe Robert, *Limit results for Markovian models of TCP*, Proceedings of IEEE Globecom, San Antonio, Texas, USA, Nov. 2001.

[10] Mihaela Enachescu, Yashar Ganjali, Ashish Goel, Nick McKeown,Tim Roughgarden *Routers with very small buffers*, ACM/SIGCOMM Computer Communication Review, Volume 35, Number 3, July 2005

[11] Mihaela Enachescu, Yashar Ganjali, Ashish Goel, Nick McKeown, and Tim Roughgarden. *Routers with very small buffers*. In Proceedings of the IEEE INFOCOM'06, Barcelona, Spain, April 2006.

[12] Azeem Feroz, Shivkumar Kalyanaraman, Amit Rao, *A TCP-friendly traffic marker for IP differentiated services*, Eighth International Workshop on Quality of Service IWQOS, 2000.

[13] Filliben, J.J., *The probability plot correlation coefficient test for normality*, Technometrics, 17(1), 1975. See also
$http://mathworld.wolfram.com/Quantile-QuantilePlot.html$

[14] Guohan Lu, Xing Li. *Modeling TCP Window Evolution Process with both Discrete and Fluid Models*. Technical report, Dept. of Electrical Enginnering Tsinghua University, Beijing, P.R. China

[15] Joao P. Hespanha, Stephan Bohacek, Katia Obraczka, Junsoo Lee. *Hybrid Modeling of TCP Congestion Control*. Hybrid Systems: Computation and Control : 4th International Workshop, HSCC 2001 Rome, Italy, March 2001.

[16] T. V. Lakshman, U. Madhow, and B. Suter, *Window-based Error Recovery and Flow Control with a Slow Acknowledgment Channel: a Study of TCP/IP Performance*, Proceedings of INFOCOM '97, Kobe Japan, April 1997.

[17] Lilliefors, H. (June 1967), *On the Kolmogorov-Smirnov test for normality with mean and variance unknown*, Journal of the American Statistical Association, Vol. 62. pp. 399-402. See also
$http://en.wikipedia.org/wiki/Lilliefors\_test$

[18] M. Mitzenmacher and R. Rajaraman. *Towards More Complete Models of TCP Latency and Throughput*, Journal of Supercomputing, 2001.

[19] T. Ott, J.H.B. Kemperman, and M. Mathis, *The Stationary Behavior of Ideal TCP Congestion Avoidance*, Website draft, August 1996
$http://citeseer.ist.psu.edu/ott96stationary.html$

[20] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, *Modeling TCP Reno Performance: a Simple Model and its Empirical validation*, IEEE/ACM Transactions on Networking, Vol. 8, no. 2, April 2000, pp. 133-145.

[21] Hidenari Sawashima, Yoshiaki Hori ,Hideki Sunahara , *Characteristics of UDP Packet Loss: Effect of TCP Traffic*, Proceedings of INET'97, June 1997

[22] R. Shorten, F. Wirth, and D. Leith, *A positive systems model of TCP-like congestion control: Asympototic results*, Hamilton Institute, Tech. Rep. 2004-1, April 2004

[23] Fredrik Solsvik, Stian Michaelsen, Arne Oslebo and Peder J. Emstad, *Live measurements on a single packet stream over a congested link from the NTNU campus*, Q2S Centre of Excellence, Department of Telematics, NTNU and UNINETT, Trondheim

[24] W. Richard Stevens *TCP/IP Illustrated, Volume 1*, Addison Wesley Longman, 1994.

[25] Gaurav Raina, Damon Wischik, *Buffer sizes for large multiplexers:TCP queueing theory and instability analysis*, EuroNGI conference on Next Generation Internet Networks, 2005

[26] Gaurav Raina,Don Towsley,Damon Wischik, *Part II: Control Theory for Buffer Sizing*, ACM/SIGCOMM CCR 2005

[27] Phuoc Tran-Gia,Member IEEE, and Hamid Ahmadi, Member IEEE. IBM Research Division. *Analysis of a Discrte Time Queeing System with Batch Arrivals and its Applications in Packet-Switching Systems*, Technical Report No. 1730, August 1988.

[28] C. Villamizar and C. Song. *High performance tcp in ansnet.* ACM Computer Communications Review, 1994.

[29] Zabell, S. L. *Alan Turing and the Central Limit Theorem.* Amer. Math. Monthly 102, 483-494, 1995.

[30] Internet reference, $http://nsnam.isi.edu/nsnam/index.php/User\_Information$

Appendix A

Queue distribution using G/D/1K discrete analysis

We bring a short summary of the $G/D/1/K$ iterative algorithm for finding the Q distribution, according to [27], using the original notations. Notations are as follows:

- $x_n(k)$ - probability that the nth arrival was equal to k packets. Using our notation it is simply the distribution of the total arrival rate R(k)
- $a_n(k)$ - distribution of the time interval between the arrival n and $n+1$. We assume that this parameter is constant (zero for all k, but for one value of k , for which it is equal to 1. For example if this $k = 10$ then the arrivals happen deterministically every 10 time units.
- $u_n(k)$ - probability to be in state $Q = k$ at the n-th iteration immediately prior to the n-th arrival
- $u_n^+(k)$ - probability to be in state $Q = k$ at the n-th iteration immediately after the n-th arrival

The iterative algorithm consists of two steps which are being run every iteration:

step 1

$$\begin{aligned} u_n^+(k) &= \sum_{j=0}^{k} u_n(j)x_n(k-j) \quad \text{k=0,1,....,B-1} \\ u_n^+(B) &= 1 - \sum_{j=0}^{B-1} u_n^+(j) \qquad \text{k=B} \end{aligned} \tag{A.0.1}$$

step 2

$$\begin{aligned} u_{n+1}(k) &= \sum_{j=k}^{B} u_n^+(j) + a_n(j-k) && \text{k=1,....,B} \\ u_{n+1}(0) &= 1 - \sum_{j=1}^{B} u_{n+1}(j) && \text{k=0} \end{aligned} \tag{A.0.2}$$

The equilibrium state probabilities converge to:

$$u(k) = \lim_{n \to \infty} u_n(k) \tag{A.0.3}$$

We tested this algorithm with queues of various buffer sizes. The complexity of convergence depends on the size of the buffer B and the desired precision, and may typically take less then 100 iterations as well as more than 1000.