Modelling of Packet Switched Network Over TCP/IP Suggest to Modify Routing to Reduce Window Based Congestion Control.

Baswaraj[#], Dr.K Prasada Rao[*], Jyothi Noubade[#]

[#] *Research Scholar Department of Computer Science, Mewar University, Chittorgarh*

Professor Department of MCA,CMR Institute of Technology,Bangalore-37

Sr.Software Engineer,CTS Pvt.Ltd,Bangalore-43

[1]*basawarajnb@gmail.com*

[2.] *kprasudu@yahoo.com*

[3]*jyothinoubade12@gmail.com*

Abstract— This paper presents a model for the ACK-clock inner loop, common to virtually all Internet congestion control protocols, and analyses the stability properties of this inner loop, as well as the stability and fairness properties of several window update mechanisms built on top of the ACK-clock. Aided by the model for the inner-loop, two new congestion control mechanisms are constructed, for wired and wireless networks.

Internet traffic can be divided into two main types: TCP traffic and real-time traffic. Sending rates for TCP traffic, e.g., file-sharing, uses window-based congestion control, and adjust continuously to the network load. The sending rates for real-time traffic, e.g., voice over IP, are mostly independent of the network load. The current version of the Transmission Control Protocol (TCP) results in large queuing delays at bottlenecks, and poor quality for real-time applications that share a bottleneck link with TCP.

The first contribution is a new model for the dynamic relationship between window sizes, sending rates, and queue sizes. This system, with window sizes as inputs, and queue sizes as outputs, is the inner loop at the core of window-based congestion control. The new model unifies two models that have been widely used in the literature. The dynamics of this system, including the static gain and the time constant, depend on the amount of cross traffic which is not subject to congestion control. The model is validated using ns3 simulations, and it is shown that the system is stable. For moderate cross traffic, the system convergence time is a couple of roundtrip times. When introducing a new congestion control protocol, one important question is how flows using different protocols share resources

Keywords:-congestion, tcp.ip, window, ack, queue

I. INTRODUCTION

The congestion control is an Internet workhorse. Up to 90% of the traffic is managed by the TCP protocol, used for web browsing, file- sharing, email transmission, and innumerable other applications. The remaining traffic serves applications such as voice over IP, online gaming and Domain Name System (DNS) service. To a first approximation, Internet traffic can be divided into two types: TCP traffic and real-time traffic. The TCP traffic uses window-based congestion control, which adapts each flow's average sending rate to the flow's fair share of available resources. The collection of TCP flows tries to use all available capacity, but it also responds to network congestion, and will reduce each flow's sending rate if the network load increases, or if the capacity in the network is reduced.

The real time traffic uses different mechanisms for congestion control, or none at all, together with explicit or implicit admission control. Which means that the sending rate of a real-time application is mostly independent of the current network state? The reason that this real-time application has, so far, not caused any Internet breakdown, is in part due to the admission control, and in part due to the fact that the typical real-time application does not need high capacity. In the current Internet, the quality of real-time applications degrades severely if they share a bottleneck with TCP flows.

This paper is organized as follows. First, in Sec. 1.1, we look into the future, to see what the Internet will be like, and what will be needed to get there. Sec. 1.2 gives a brief introduction to IP networking and the window-based congestion control of TCP. In Sec. 1.3, we describe the control objectives for Internet congestion control. An overview of the contributions in this thesis is given in Sec 1.4. Publications are listed in Sec. 1.5, and finally, Sec. 1.6 provides the outline for the rest of the paper.

a. An emergency response scenario:

Mumbai, November 2015. A week ago, the embankments around Mumbai and the city center broke down; they could

not hold against the severe autumn storm, and the higher sea level we have seen in recent years. Since the sea water flooded several underground tunnels, transformer stations, telephone switches, and Internet peering centrals, the basic infrastructure is still not working (not to mention the flooded sewer system).The city centre, roughly from Dadar up to Andheri, is flooded. Still, the area is far from deserted, a few hundred people are working there, local emergency response personnel, additional professionals from Bandra and Matunga, to gather with volunteers who live or work in the area and decided to stay and help.

The canter of all this activity is the well-known VSNL building (Fig 1.1). When the electricity and the telephone system stopped working, the people working in one of the offices half-way up in the tower realized that the location was ideal for wireless communication to large parts of the flooded area. Enabling ad-hoc networking, they soon got connected to other people living or working nearby, who were also cut -off from the outside world.

The network grew to about 150 active nodes, and soon somebody managed to get in contact with emergency response people Moreover, as the VSNL building was the worst-affected, the networks of the entire nation had a shutdown, owing to the natural calamity. Within an hour, a military helicopter landed on the roof. The passengers were a couple of emergency response workers from Bandra fire station, who brought with them a microwave link and a small diesel-powered generator, and sufficient fuel for a couple of days. As soon as the microwave was operational, the ad-hoc network had a reasonable connection to the Internet. The network was used primarily for voice calls and email, first locally, and, once there was connectivity to the outside, to the whole world. People could get in contact with relatives, and the emergency response people could get in contact with the people in the area to assess the situation and organize reinforcements.

The next few days, things went smoothly. Evacuation, by boat or helicopter when necessary, was organized. Patients waiting for evacuation were equipped with networked pulse oximeters and other sensors, monitored both by local medical staff and the medical support canter at the Mumbai University Hospital. The coordination office moved down to the bottom of the tower, since it proved difficult to use the temporary generator to power the tower's elevators. The people from Bandra and Matunga arrived, and they started to work, almost not noticing how their computers automatically joined the communication network. The support for those who stayed was organized, based around the Telephone Exchange tower: Meals, temporary hygiene facilities, as well as electrical charging stations for everybody's computers. Food and fuel were delivered by boat from managers.

The demands on the network changed gradually during this week. The whole time, voice and email were the most important applications, but there were also sensor data, transfer of photos and videos documenting the situation, and file-sharing of all sorts of useful information, including large files with detailed blueprints for buildings and infrastructure. This is the first time in India that Internet communication has played such an important role in an emergency situation, but it is not unique in the world.

b. Network Layer:- A communication network is a set of links and nodes. The set of nodes can be divided further into routers, which forward packets between links, and hosts, which are the communication end points. On the physical layer, common link types are based on optical fibers, copper cables, or radio transmission. When more than one node can transmit on the same link, the use of the link is coordinated using some medium access control. This coordination, together with the conventions for how to encode an IP-packet for a particular link type, is referred as the link layer. The service provided by the link layer is the transmission of IP packets between nodes that are connected to the same link

The IP network is implemented on top communications links based on many different link technologies. Above IP, there are a couple of widely used transport protocols, and an innumerable number of different applications. To provide global connectivity, forwarding of packets between links is necessary. Deciding which path each packet should take between its source and destination is the responsibility of the routing system. Routing, together with the addressing architecture, are the main services of the network layer. The network layer provides best effort delivery of IP packets to and from arbitrary nodes in the network.

c.Window based congestion control:- TCP uses a sliding window flow control. The window limits the amount of data that can be sent without waiting for Acknowledgement (ACK) from the receiver. When the window is constant, these results in the so-called "ACK-clock"; the timing of each sent packet is determined by the reception of the ACK for an earlier packet. This is illustrated in Fig. 1. earlier. Window-based congestion control was originally motivated by the fluid flow analogy. The packet flow is what a physicist would call 'conservative': A new packet isn't put into the network until an old packet leaves. The physics of flow predicts that systems with this property should be robust in the face of congestion. Van Jacobson, [59]

One can think about the sliding window and the ACK clock as a peculiar inner control loop, which determines the sending rate; when the Roundtrip Time (RTT) fluctuates, the sliding window gives an average sending rate of one full window per average RTT. The window size is adjusted depending on received ACKs, and it is the details of this outer-loop that differ between TCP variants. The most widely used version of TCP is New Reno, which uses the following window update rules. As long as there are no packet losses,

the window size is increased by one packet per RTT. And when a packet is lost, the window size is reduced to one half of its previous value. The linear increase of the window size, in the absence of packet losses, together with the multiplication by 1/2 when a packet loss is detected, is called Additive Increase, Multiplicative Decrease (AIMD).

This paper proposes a new model for the queuing dynamics under window- based congestion control. The dynamic behaviour depends on the amount of real- time, non-congestion controlled, cross-traffic that the network is shared with. The main novelty in the model is to accurately describe the influence of this cross-traffic, and to note that this influence is significant; in terms of central system properties such as gain and time constants.

The most important concept in TCP congestion control is that of the congestion window. The window is the amount of data that has been sent, but for which no Acknowledgement has yet been received. A constant congestion window means that one new packet is transmitted for each ACK that is received.

The sending rate is controlled indirectly by adjusting the congestion window. The standard way of doing this is called New Reno [5, 45]. It is described in this section. One common extension is TCP with selective Acknowledgements (sack) [88, 18]. Before explaining the control mechanisms, we have to look into how TCP detects packet loss.

Acknowledgement and Loss detection:

At the receiving end, Acknowledgement packets are sent in response to received data packets. TCP uses cumulative Acknowledgements: Each Acknowledgement includes a sequence number that says that all packets up to that one have been received. Equivalently, the Acknowledgement identifies the next packet that the receiver expects. When packets are received out of order, each received packet results in an acknowledgement, but they will identify the largest sequence number such that all packets up to that number has been received.

For example, if packets 1, 2, 4, and 5 are received, four Acknowledgements are generated. The first says, "I got packet #1, I expect packet #2 next", while the next three Acknowledgements all say, "I got packet #2, I expect packet #3 next".

The last two Acknowledgements are duplicate ACKs, since they are identical to some earlier ACK. When duplicate ACKs are observed on the sender side, there are several possible causes: a packet may have been delayed and delivered out-of-order, a packet may have been lost, or an ACK packet may have been duplicated by the network. Packet losses are detected by the sender in two ways:

1. **Timeout**: If a packet is transmitted and no ACK for that packet is received within the Retransmission Timeout interval (RTO), the packet is considered lost.
2. **Fast retransmit:** If three duplicate ACKs are received, the "next expected packet" from these ACKs is considered lost. Note that this cannot happen if the congestion window is smaller than four packets.

Packets that are lost, as detected by either of these mechanisms, are retransmitted. Furthermore, congestion control actions are also based on these loss signals, as described below. The value for RTO is not constant, but based on measured average and variation of the RTT. It is also modified when a timeout occurs, by the exponential back-off mechanism.

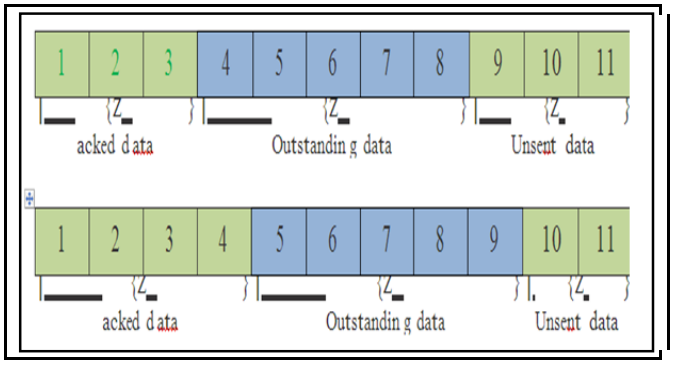

Figure 1. Sliding window control. The window size is the amount of outstanding data.

TCP congestion control state:

There are four distinctive states in the TCP congestion control, illustrated in Figure 3, and two state variables related to congestion control: The congestion window cwnd and the slow start threshold ssthresh. The value of ssthresh determines the window increase rule; if cwnd < ssthresh, TCP increases cwnd by one packet for each received ACK (slow start state), and if cwnd ≥ ssthresh, TCP increases the window size by one packet per RTT (congestion avoidance state). Typical initial Values when TCP leaves the idle state and enters the slow start state are a cwnd of 2 packets and an infinite ssthresh. We look at the operation of each of the four states in turn.

Slow start: The slow start state is the first state entered when a flow is created, or when a flow is reactivated after being idle. The slow start state can also be entered as the result of a timeout. In this state, Cwnd is increased by one packet for each non-duplicate ACK.

The effect is that for each received ACK, two new packets are transmitted. This implies that the congestion window, and also the sending rate, increases exponentially, doubling once per RTT.It may seem strange to refer to an exponential increase of the sending rate as "Slow start"; the reason is that in the early days, TCP used a large window from the transitions back to the idle state are omitted.

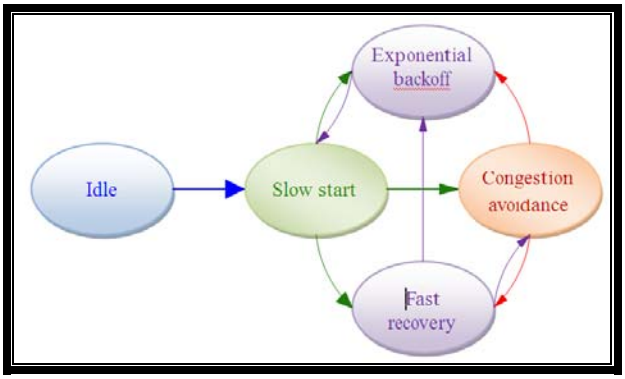

Figure 2. Start and the introduction of the slow start mechanism did slow down connection startup. Slow start continues until either

- Cwnd > ssthresh, in which case TCP enters the congestion avoidance state, or
- A timeout occurs, in which case TCP enters the exponential back-off state, or
- Three duplicate ACKs are received, in which case TCP enters the fast recovery state.

The motivation for the slow start state is that when a new flow enters the network, and there is a bottleneck link along the path, then the old flows sharing that link need some time to react and slow down before there is room for the new flow to send at full speed.

d. Problem statement:

The overall goal of congestion control is to optimize the performance in a com- medication network. This optimization means, roughly, that sending rates at the data sources should be as high as possible, without overloading the network. The primary measure of network overload is packet losses; when the arrival rate at a link exceeds capacity, the corresponding queue starts to build up, and when the queue is full, packets must be discarded. The bottleneck links in the network should be fully utilized. The requirement of a small loss rate implies that the average arrival rate at each bottleneck link should either match the link capacity exactly, or be very slightly larger.

When the network is shared with real-time traffic, it becomes important not only to maintain a small packet loss rate, but also to maintain reasonably small queues, since large queues imply large delays. By the term congestion control, we include both the rules for adjusting window sizes and sending rates, which are implemented in end nodes, and all related rules and mechanisms, such as Active Queue Management schemes (AQM) and Explicit Congestion Notification (ECN), which are implemented in routers.
Control Objective: Congestion control has been a very active area of research during the last decade. Still, the current transport control protocol TCP works fairly well. So why change that which is not broken? To understand that, we must first understand what the control objectives are:

Avoid network overload: The probability of packet loss due to congestion should **be** small.

Efficient resource utilization: All bottleneck links should be fully utilized.

Fair sharing: The congestion control mechanism determines how resources are shared between users. The sharing should be predictable and satisfy some reasonable notion of fairness.

React to changes: The network load is constantly varying, there are occasional routing changes, and some (wireless) links have varying capacity and delay. The congestion control must react to these changes and adjust the sending rates.
Small queuing delays: Both large queues and large queue fluctuations can harm other applications using the network, in particular real-time applications of these objectives, TCP achieves all but the last one, when used over the ordinary wired network for which it was designed.

For low-power wireless links, and for links with large propagation delay or large bandwidth-delay product, TCP also has difficulties achieving efficient resource utilization. We therefore see three main problems that are being addressed by current TCP research:

TCP over wireless: TCP interprets packet losses as a sign of congestion. If a wireless link loses some packets due to disturbances on the radio channel, TCP Reduces its sending rate even when there is no real congestion. Many wireless links tries to avoid this problem by retransmitting lost packets at the link layer. However, this leads to larger delays and larger delay variations, which are also problematic for TCP.

High capacity, large delay networks: To use available capacity efficiently, TCP needs a window size close to or larger than the bandwidth-delay product. In the basic TCP protocol, the window size is limited to 64 Kbyte, which is quite small with today's high-speed links.

A 100 Mbit/s link with 5 ms RTT has a bandwidth-delay product of 61 Kbyte. To overcome this problem, one can use the window scaling option [60]. But even when window scaling is used, so that the window size can grow to match the bandwidth-delay product, the AIMD rules imply that it takes a large number of RTTs for the window size to grow large enough. If also the RTT is large, this leads to poor throughput.
II. Modelling: In this research we develop and validate an improved model for the feedback system consisting of a

queuing network, and data sources using window-based congestion control. The window sizes are the system inputs, and the queue sizes are the system outputs. This model describes the inner-loop of all TCP-like window-based congestion control methods. The main novelty of the model is that it takes into account that the network is shared with traffic, which is not subject to congestion control, such as video streaming, and voice over IP. The amount of such cross-traffic has a large influence on the dynamics of the system, in particular the gain and the time constant.

A step response is shown in Fig 3.. The dashed curves represent two models, the static model [114] and the integrator model [56, 80], that have been used in the literature. The dotted curve is the result from ns3 simulations, and the solid curve is the new model. The amount of cross-traffic is an important parameter for the dynamics; in this figure, there is 30% cross-traffic.

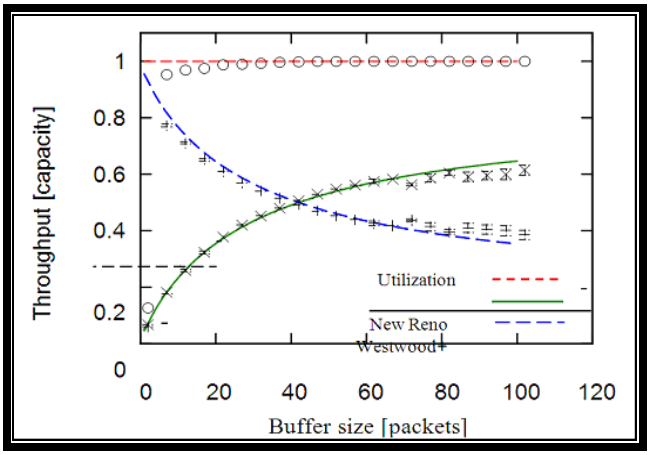

Figure 3. Normalized throughput for TCP New Reno (increasing curve) and West- wood+ (decreasing curve), as a function of the buffer size at the bottleneck router. The topmost almost flat curve is the link utilization.

III.Analysis: The stability properties of the inner-loop in window-based congestion control, as well as the convergence time. Analyses the fairness when flows using two different flavours of TCP, New Reno and Westwood+, compete for bottleneck capacity.

This will also give us an opportunity to look closer at how AIMD works, and how it relates to buffer sizing issues. As seen in Fig 1, the sharing depends on the buffer size at the bottleneck. The two flows share the link equally only when the buffer size equals the bandwidth-delay product, which in this case are 42 packets.

IV.Design:- we design a new congestion control mechanism, as an outer-loop around the inner-loop that was modelled earlier. The aim is to maintain the efficiency and fairness properties of TCP, but with significantly smaller bottleneck queues.

The key ideas are to take advantage of the stability of the inner-loop, and to use rules for setting and reacting to packet marks that results in more frequent feedback than with AQM and ECN. Figure 4. Shows the response in window size and queue size to a change in the cross-traffic intensity, for TCP New Reno and for the new mechanism.

V.Congestion Control Avoidance:

In the congestion avoidance state, cwnd is increased by one packet per RTT (if cwnd reaches the maximum value, it stays there). This corresponds to a linear increase in the sending rate. On timeout, TCP enters the exponential back-off state, and on three duplicate ACKs, it enters the fast recovery state. The motivation for this congestion avoidance mechanism is that since TCP does not know the available capacity, it has to probe the network to see at how high a rate data can get through. Aggressive probing would make the system unstable, and a single packet increase seems to work well in practice.

Exponential back-off:

TCP enters the exponential back-off mode after timeout. Several actions are taken when entering this state:

- The lost packet is retransmitted.
- The state variables are updated by ssthresh ← cwnd/2, cwnd ← 1 packet.
- The RTO value is doubled.

When an ACK for the retransmitted packet is received, TCP enters the slow start phase. The above update of ssthresh implies that, in the absence of further packet losses, TCP will switch from the slow start phase to the congestion avoidance phase once cwnd is increased to half its value before the timeout. If the retransmission timer expires again with no ACK for the retransmitted packet, the packet is repeatedly retransmitted, RTO is doubled, and ssthresh is set to 1 packet [40]. The upper bound for the RTO is on the order of one or a few minutes. Exponential back-off continues until an Acknowledgement for the packet is received, in which case TCP enters the slow start phase, or the TCP stack or application gives up and closes the connection.

The motivation for the exponential back-off mechanism is that timeouts, in particular repeated timeouts are a sign of severe network congestion. In order to avoid congestion collapse, the load on the network must be decreased considerably and repeatedly, until it reaches a level with a reasonably small packet loss probability.

In this example, it is five packets. In the top figure, eight packets have been transmitted, and ACKs for the first three packets have been received. The above figure illustrates the situation after the ACK for the fourth packet has been received.

One more packet, the ninth, is transmitted, and the window of outstanding data “slides” forward one packet. Reassemble the stream at the other end, and to detect if packets are lost,

reordered or duplicated by the network. The protocol also implements flow control, which prevents the sending node from overloading the receiver, and congestion control, which prevents the sending node from overloading the network. In a recent study of the traffic mix in BSNL's ADSL network, more than 90% of the traffic was using TCP and a significant proportion thereof was peer-to-peer file-sharing.

Everything on top of the transport layer is referred to as application layer. There are innumerable application protocols, from the older telnet protocol and Simple Mail Transfer Protocol (SMTP), to the Hypertext Transfer Protocol (HTTP) used for the World Wide Web and the Session Initiation Protocol (SIP) used to set up IP telephony

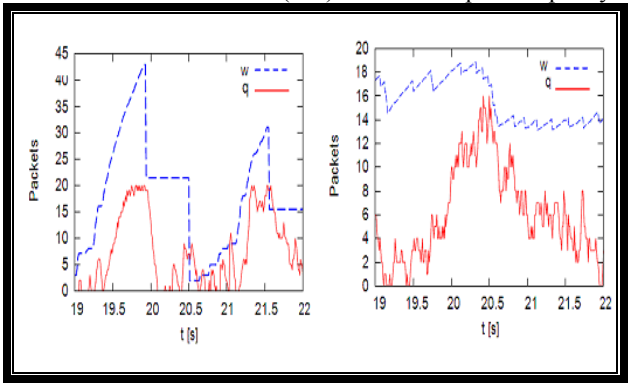

Figure 4. Window size (w) and bottleneck queue size (q) for a short segment of a simulation with a single bottleneck and a congestion controlled flow sharing the bottleneck with Poisson cross-traffic. At t = 20 s, the cross-traffic intensity is increased from 20% to 40% of link capacity. To the left: New Reno. To the right: The proposed mechanism. Note the different scale on the vertical axis.

VI.Mathematical Models for Congestion

One crucial issue in model-based control design is to select a model with the right level of detail. The model should capture the important dynamics of the real system, and at the same time be simple enough for mathematical analysis of the system.

The models for congestion control, and related tools, can be classified as:

- Packet-level: A packet-level model accounts for the location of each individual packet, as the packets are queued and forwarded by the network. The system state evolves as a series of discrete events, where the events of interest are arrival and departure of packets, and protocol timeouts.

- Fluid flow: A fluid flow model sees the data transport as a continuous fluid, with no packet boundaries. State variables vary continuously, and are described using differential equations. In the context of congestion control, fluid flow models usually do not try to capture all details of the dynamics. Instead, the state variables represent averages of the true system state, where the average can be an average over an RTT, or an expected value with respect to stochastic features of the system.

- Hybrid: In a hybrid model, the evolution of the state is a result of discrete events, together with continuous changes between events. Typically, a continuous model is used for the queuing dynamics and maybe for some of the end-host protocol actions, while protocol actions such as the multiplicative decrease in TCP is modelled as a discrete event, and a corresponding discontinuous state update.

In this section, we will describe these in turn. We will also describe the framework of network utility maximization, which is an important tool for analysis and understanding of large networks.

Packet Level Models: For telecommunication networks, there is a long tradition of using queuing and traffic theory, going back to the work of Erlang in the early 20th century [41]. These are tools for analysing quality measures such as queuing delay and blocking probability, given capacity and demand. The theory gives particularly nice results when the arrival of traffic is a Poisson process. A later development is adversarial queuing theory [21], where the arrival traffic is not stochastic, but chosen by an adversary, subject to load constraints. Web-server admission control is studied in [99], with the central trade-off between request rejection rate, and response time of the admitted requests.

The performance, in terms of blocking probability and delay, of a single server with several traffic classes, is analysed in [4]. In [103], it is argued that traffic theory is an essential tool for developing an Internet with multiple service classes. Still, queuing and traffic theory is best suited for analysis of mechanisms related to quality-of-service and admission control, which are of an open loop character, and more difficult to apply to closed-loop congestion control. A recent application to TCP/AQM-style congestion control is found in [53].

The properties of interconnected queuing systems, in particular when there are several service classes, are remarkably complex. A recent preprint [47] proves that for queuing networks with multiple service classes, a given deterministic arrival process, and given deterministic service times, stability is an undividable problem. I.e., there exists no algorithm that takes as input a description of a queuing network, an arrival process, and an initial state, and determines whether or not the queue lengths of the system are bounded over time.

VII.Fluid Flow Model

As the name implies, fluid-flow models do not try to describe individual packets, but view the data streams across the network as continuous flows. All the quantities of interest, such as sending rates, window sizes, and queue sizes, are treated as continuous (and often differentiable) functions of

time. We start with the fluid-flow model for a queue.

Example 2.1 (Queue dynamics)

Assume that we have packets arriving to a queue as a Poisson process with time-varying intensity r(t)/m, where r(t) is Continuous and m is the packet size. Also assume that the service times are exponentially distributed with parameter m/c, independent of the arrival process. This is an M/M/1 queue with arrival rate r(t)/m and service rate c/m. Let N (t) denote the number of packets in the queue at time t. The amount Of queued data is then q(t) = mN (t). We now fix some t, with q(t) > 0, and Examine the change of the queue size during the interval [t, t + h].

Let Ah be the number of arriving packets during this interval, then Ah is Poisson distributed with parameter Rt+h r(s)/m. Let S be the number of departing packets during the same interval. For small h, we can ignore the possibility that the queue becomes Empty and approximate Sh as Poisson distributed with parameter hc/m. The change in the queue size is then

VII. Conclusion: Through the chapters of this thesis, we have developed and validated a new model for the window-based packet transmission which is the inner-loop of all window- based congestion control protocols. We have analyzed the properties of this system, and a couple of different outer-loops, in the form of window update rules. Finally, we have designed two new outer-loops, one aimed for general congestion control, and one specialized to cellular systems where some extra cross-layer signaling is possible. So what can we learn about modeling, analysis, and design, of window-based congestion control.

Feature Enhancement: What the Internet traffic mix will be like in the future is hard to predict. Maybe real-time traffic will remain a small fraction, less than 10%, of the Internet traffic? Or maybe real-time video conferencing (which certainly is more environmentally friendly than international meetings with the corresponding long-distance traveling) will be the next killer application? Or some other unforeseen real-time application that needs much higher capacity than today's voice over IP and online games. The traffic mix can certainly be of different character in different parts of the Internet. When designing congestion control protocols, it is therefore important to take into account the possibility of a large proportion of the traffic being real-time traffic. One prediction that it seems quite safe to make, is that we will see more and more devices and users that are connected to the Internet via wireless links. As devices get cheaper, the laptops and mobile phones today carried by humans will be joined by wireless sensors for various industrial applications as well as environmental monitoring.

References:

[1].3rd Generation Partnership Project; Technical specification group radio Access network; physical layer procedures (FDD), 2004. TS 25.214 V6.1.0.

[2].3rd Generation Partnership Project; Technical specification group radio access network; High Speed Downlink Packet Access (HSDPA), overall description, 2006. TS 25.308 V7.0.0.

[3]. A. A. Abouzeid, S. Roy, and M. Azizoglu. Stochastic modeling of TCP over loss links. In INFOCOM (3), volume 3, pages 1724–1733, 2000.

[4]. S. Ahmad, I. Awan, and B. Ahmad. Performance modeling of finite capacity queues with complete buffer partitioning scheme for bursty traffic. In First Asia International Conference on Modeling & Simulation, pages 264–269,2007.

[5].M. Allman, V. Paxson, and W. Stevens. TCP congestion control. RFC 2581, April 1999.

[6].T. Alpcan and T. Basar. A globally stable adaptive congestion control scheme for Internet-style networks with delay. IEEE Wireless communications, 12(6): 42–49, December 2005.

[7].E. Altman, C. Barakat, S. Mascolo, N. Mo¨ller, and J. Sun. Analysis of TCP Westwood+ in high speed networks. In International Workshop on Protocols for Fast Long-Distance Networks, Nara, Japan, January 2006.

[8]. Y. H. Aoul, A. Mehaoua, and C. Skianis. A fuzzy logic-based AQM for real- time traffic over Internet. Computer networks, 51(16):4617–4633, November 2007.

[9].G. Appenzeller, I. Keslassy and N. McKeown. Sizing router buffers.In SIGCOMM, Portland, September 2004. ACM.

[10[. S. Athuraliya, V. H. Li, S. H. Low, and Q. Yin. REM: active queue management. IEEE Networking, 15(3):48–53, 2001.

[11]. K. Avrachenkov, U. Ayesta, E. Altman, P. Nain, and C. Barakat. The effect of router buffer size on the TCP performance. In Proceedings of LONIIS workshop, 2002.

[12].K. Avrachenkov, U. Ayesta, and A. Piunovskiy. Optimal choice of the buffer size in the Internet routers. In IEEE Conference on Decision and Control and European Control Conference, Seville, 2005. IEEE CSS.

[13].F. Baccelli and D. Hong. TCP is max-plus linear: and what it tells us on its throughput. In SIGCOMM, pages 219–230, 2000.

[14].F. Baccelli and D. Hong. AIMD, fairness and fractal scaling of TCP traffic.In Proceedings of IEEE INFOCOM, pages 229–238, New York, June 2002.

[15].H. Balakrishnan, S. Seshan, E. Amir, and R. Katz. Improving TCP/IP per- formance over wireless networks. In Proc. 1st ACM Conference on Mobile Communications and Networking (Mobicom). ACM, Berkeley 1995.

[16].C. Barakat and E. Altman. Bandwidth tradeoff between TCP and link-level FEC. Comput. Networks, 39(5):133–150, 2002.

[17].D. Barman, I. Matta, E. Altman and R. El Azouzi. TCP optimization through FEQ, ARQ and transmission power tradeoff. In International Con- ference on Wired/Wireless Internet Communications WWIC, february 2004.

[18].E. Blanton, M. Allman, K. Fall, and L. Wang. A conservative selective ac- knowledgment (SACK)-based loss recovery algorithm for TCP. RFC 3517, April 2003.
[19].S. Bohacek, J. P. Hespanha, J. Lee, and K. Obraczka. Analysis of a TCP hybrid model. In Annual Allerton Conference on Communication, Control, and Computing, 2001. Extended version available at http://www.ece.ucsb. edu/hespanha/techrep.html.
[20]. J. Border, M. Kojo, J. Griner, G. Montenegro, and Z. Shelby. Performance enhancing proxies intended to mitigate link-related degradations. RFC 3135, June 2001.
[21] Allan Borodin, Jon Kleinberg, Prabhakar Raghavan, Madhu Sudan, and David P. Williamson. Adversarial queueing theory. In Proceedings of the twenty-eighth annual ACM symposium on Theory of computing, pages 376 385, 1996.
[22] J.-Y. Le Boudec and P. Thiran. Network Calculus: A Theory of Deterministic Queuing Systems for the Internet. LNCS. Springer, 2001.
[23] Stephen Boyd and Lieven Vandenberghe. Convex Optimization. Cambridge University Press, 2004.
[24] L. S. Brakmo and L. L. Peterson. TCP Vegas: end-to-end congestion avoidance on a global Internet. IEEE Journal on Selected Areas in Communications, 13(8):1465–1480, 1995.
[25] B. Briscoe. Flow rate fairness: Dismantling a religion. Computer communication review, 37(2):65–74, April 2007.
[26] R. Bush and D. Meyer. Some Internet architectural guidelines and philosophy. RFC 3439, December 2002.
[27] G. Carneiro, J. Ruela, and M. Ricardo. Cross-layer design in 4G wireless terminals. IEEE Wireless Communications, 11(2):7–13, April 2004.
[28] S. Cen, P. C. Cosman, and G. M. Voelker. End-to-end differentiation of congestion and wireless losses. IEEE/ACM Trans. on Networking, 11(5):703-717, 2003.
[29] T. Chahed, A-F. Canton and S-E. Elayoubi. End-to-end TCP performance in W-CDMA/UMTS. In ICC, Anchorage, May 2003.
[30] M. Chiang, S. H. Low, A. R. Calderbank, and J. C. Doyle. Layering as optimization decomposition: A mathematical theory of network architectures. Proceedings of the IEEE, 95(1):255–312, January 2007.
[31] A. Chockalingam, A. Zorzi, and V. Tralli. Wireless TCP performance with link layer FEC/ARQ. In IEEE ICC, pages 1212–1216, June 1999.
[32] J. Y. Choi, K. Koo, J. S. Lee, and S. H. Low. Global finite-time convergence of TCP Vegas without feedback information delay. International journal of control automation and systems, 5(1):70–78, February 2007.
[33] C. Chrysostomou, A. Pitsillidesa, L. Rossidesa, M. Polycarpoub, and A. Sek- erciogluc. Congestion control in differentiated services networks using fuzzy- RED. Control Engineering Practice, 11(10):1153–1170, October 2003.
[34] R. M. Corless, G. H. Gonnet, D. E. H. Hare, D. J. Jeffrey, and D. E. Knuth. On the Lambert W function. Advances in Computational Mathematics, 5:329–359, 1996.
35] Thomas M. Cover and Joy A. Thomas. Elements of information theory. Wiley, 1991.
[36] Rene L. Cruz. A calculus for network delay, part I: Network elements in isolation. IEEE Transactions on Information Theory, 37(1):114–131, 1991.
[37] Rene L. Cruz. A calculus for network delay, part II: Network analysis. IEEE Transactions on Information Theory, 37(1):132–141, 1991.
[38] A. Dhamdhere, H. Jiang, and C. Dovrolis. Buffer sizing for congested Internet links. In Proceedings of IEEE INFOCOM, 2005.
[39] J. Postel (ed.). Transmission control protocol. RFC 793, September 1981. [40] R. Braden (ed.). A requirement for Internet hosts communication layers. RFC 1122, October 1989.
[41] A. K. Erlang. Telefon-ventetider. Matematisk Tidskrift, B:25–42,1920.
[42] Marco Fiorenze. Downlink TCP proxy solutions for interactive gaming over HSDPA and 3G networks. Master's thesis, KTH, September 2007.
[43] S. Floyd. HighSpeed TCP for large congestion windows. RFC 3649, December 2003.
[44] S. Floyd, R. Gummadi, and S. Shenker. Adaptive RED: An algorithm for increasing the robustness of RED's active queue management. http://www.
cir.org/floyd/papers/adaptiveRed.pdf, August 2001.
[45] S. Floyd, T. Henderson, and A. Gurtov. The NewReno modification to TCP's fast recovery algorithm. RFC 3782, April 2004.
[46] C. P. Fu and S. C. Liew. TCP veno: TCP enhancement for transmission over wireless access networks. IEEE Journal on Selected Areas in Communica- tions, 21(2):216–228, 2003.
[47] D. Gamarnik and D. Katz. On deciding stability of multiclass queueing net- works under buffer priority scheduling policies. eprint arXiv:0708.1034v1, November 2007. http://arxiv.org/pdf/0708.1034.
[48] Y. Ganjali and N. McKeown. Update on buffer sizing in Internet routers. Computer Communication Review, 36(5):67–70, October 2006.
[49] T. Gillespie. Engineering a principle: 'End-to-end' in the design of the Inter- net. Social Studies of Science, 36(3):427–457, June 2006.
[50] Daniele Girella. Downlink TCP proxy solutions over HSDPA with flow ag- gregation and user behavior. Master's thesis, KTH, September 2007.
[51] S. Gorinsky, A. Kantawala, and J. Turner. Link buffer sizing: A new look at the old problem. In Proceedings of IEEE Symposium on Computers and Communications, pages 507–514, June 2005.
[52] L. A. Grieco and S. Mascolo. Performance evaluation and comparison of Westwood+, New Reno and Vegas TCP congestion control. ACM Computer Communication Review, 34(2), April 2004.
[53] L. Guan, M. E. Woodward I. U. Awan and, and X. Wang. Discrete-time performance analysis of a congestion control mechanism based on RED under multi-class bursty

and correlated traffic. Journal of Systems and Software, 80(10):1716–1725, October 2007.
[54] Gerhard Haßlinger, Joachim Mende, Ru¨diger Geib, Thomas Beckhaus, and Franz Hartleb.Measurement and characteristics of aggregated traffic in broadband access networks. In Proceedings of Intnernational Teletraffic Congress: Managing Traffic Performance in Converged Networks, volume 4516 of LNCS, pages 998–1010, Ottawa, June 2007. Springer.
[55] J. P. Hespanha, S. Bohacek, K. Obraczka, and J. Lee. Hybrid modeling of TCP congestion control. In M. Di Benedetto and A. Sangiovanni-Vincentelli, editors, Hybrid Systems: Computation and Control, volume 2034 of Lecture Notes in Computer Science, pages 291–304. Springer-Verlag, Berlin, Germany,2001.
[56] C. Hollot, V. Mistra, D. Towsley, and W. B. Gong. A control theoretic analysis of RED. In IEEE Infocom 2001, 2001.
[57] C. V. Hollot, V. Misra, W.-B. Gong, and D. Towsley. On designing improved controllers for AQM routers supporting TCP flows. In IEEE Infocom, pages 1726–1734, April 2001.
[58] E. Hossain, D. I. Kim, and V. K. Bhargava. Analysis of TCP performance under joint rate and power adaptation in cellular WCDMA networks. IEEE Trans. on Wireless Comm., 3(3), May 2004.
[59] V. Jacobson. Congestion avoidance and control. ACM Computer Communication Review, 18:314–329, 1988.
[60] V. Jacobson, R. Braden, and D. Borman. TCP extensions for high perfor- mance. RFC 1323 May 1992.
[61] K. Jacobsson, H. Hjalmarsson, and K. H. Johansson.