Färnqvist D., Strandemar K., Johansson K.H., Hespanha J.P.:
Hybrid Modeling of Communication Networks Using Modelica
2nd International Modelica Conference, Proceedings, pp. 209-213

Paper presented at the 2nd International Modelica Conference, March 18-19, 2002,
Deutsches Zentrum für Luft- und Raumfahrt e.V. (DLR), Oberpfaffenhofen, Germany.

All papers of this workshop can be downloaded from
http://www.Modelica.org/Conference2002/papers.shtml

Hybrid Modeling of Communication Networks Using Modelica*

Daniel Färnqvist,† Katrin Strandemar,† Karl Henrik Johansson,†‡ and João Pedro Hespanha§

Abstract

Modeling and simulation of communication networks using Modelica is discussed. Congestion control in packet-switched networks, such as the Internet, is today mainly analyzed through time-consuming simulations of individual packets. We show, by developing a model library based on a recent hybrid systems model, that Modelica provides an efficient platform for the analysis of communication networks. As an example, a comparison between the two congestion control protocols is presented.

1 Introduction

The interaction with a variety of networks plays an important role in everyone's life. A growing use of networks, such as the Internet, with their widening set of services increases the demand on the control of the network. The objective is often to improve traffic throughput and to better accommodate different service demands. Communication networks experience major problems due to traffic congestion. Today's congestion control is in most networks implemented as end-to-end protocols, e.g., [4, 12, 10]. The protocols have proved to form the basis of a remarkably robust and scalable system, though the understanding of the basic principles of these complex systems is far from satisfactory. There is intensive research on modeling and simulation of the Internet. It has been pointed out that classical network models from telecommunication based on Poisson modeling are not suitable for the Internet [8]. Recent models capturing the self-similar nature of the traffic has been developed [5]. The general opinion in the network area is still that Internet modeling and simulation are open research problems [9].

*The authors want to thank Håkan Hjalmarsson and Gunnar Karlsson for helpful discussions.

†Dept. of Signals, Sensors & Systems, Royal Institute of Technology, SE–100 44 Stockholm, Sweden

‡Corresponding author. Tel. +46 8 7907321. Fax +46 8 7907329. kallej@s3.kth.se

§Dept. of Electrical & Computer Engineering, University of California, Santa Barbara, CA

The intention of this paper is to describe initial work on modeling packet-switched communication network using Modelica. The standard modeling and simulation environment targeted at networking research is the discrete event simulator `ns-2` [7]. `ns-2`, which was originally developed at UC Berkeley, implements network protocols such as the transmission control protocol (TCP) and traffic source behaviors such as file transfer protocol (FTP) and Telnet. Since `ns-2` directly implements the Internet protocols and simulates individual packets, it provides on one hand accurate simulation results but on the other hand a rather slow simulation speed. The result of this is that `ns-2` is mainly for studying relatively small networks over a short time scale. The other extreme is to use flow models, i.e., to approximate the packet transmission with a continuous flow and basically neglect the network protocols. Flow models capture average transmission rates but ignores events such as packet drops. Flow models are hence suitable for the study of steady-state behavior but not for evaluating transient phenomena. This is for instance due to that certain congestion control strategies are based on the implicit feedback information from packet drops, which are not included in the flow model. A hybrid systems model, which is based on the average rates but takes packet drops into account, was recently proposed in the literature [3]. The motivation for this model is to capture the network behavior on a time scale in between packet models and flow models. Studies have shown that the hybrid model is able to model many important network phenomena [3, 1]. In this paper, we will show that the hybrid model is suitable for Modelica. Moreover, we show that simulating the model in Dymola provides an efficient environment for studying congestion control in computer networks.

The outline of the paper is as follows. Section 2 presents a brief introduction to congestion control in communication networks. The hybrid model is described in Section 3 and its Modelica implementation is discussed in Section 4. An example, where two TCP versions are compared for a small wireless network, is given in Section 5.

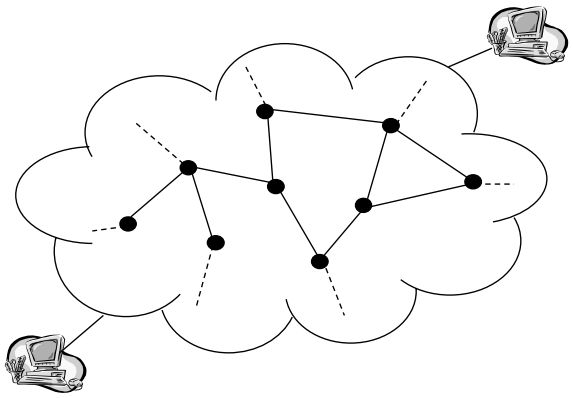

Figure 1: Communication network.

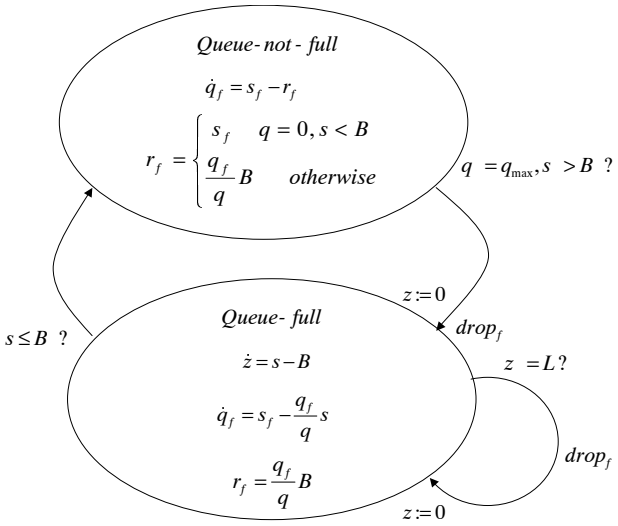

Figure 2: Hybrid queue model.

2 Communication Networks

A packet-switched communication network can be described by a directed graph. The nodes represent the routers, which direct the packets from sender to receiver, and the edges correspond to wired or wireless links. Figure 1 illustrates a connection with one sender and one receiver. The bandwidths of the links are limited, so each router has a buffer where packets are stored if more packets are entering the router than is going out. In this way, it is possible to deal with minor traffic congestion in the network. If too many packets enter a router in a short amount of time, however, packets will be dropped due to that the buffer has finite size. The way this congestion problem is handled by the senders on the Internet is through a control mechanism denoted transmission control protocol (TCP). The receiver sends acknowledgments back to the sender, when packets have arrived. In order to efficiently use the network resources, the TCP sender adjusts it sending rate according to a control variable called the window size w. The TCP sender sends w number of packets and waits for acknowledgments for them to return. Hence, w corresponds to the number of unacknowledged packets the sender may have in the network. When the sender has received acknowledgments for all w packets, w is increased by one. If a packet is dropped (so that no acknowledgment for that packet is received), w is decreased by a factor two. Hence, TCP uses additive increase and multiplicative decrease (AIMD) to regulate the congestion window size based on explicit acknowledgments and implicit negative acknowledgments. Although, the AIMD control strategy has proved to be efficient, robust and remarkably scalable for the Internet, it is believed that it might be too abrupt for emerging applications such as streaming of audio and video.

3 Hybrid Model

In the hybrid model for communication networks proposed by Hespanha et al. [3, 1], the network dynamics and the TCP dynamics is modeled as hybrid systems (e.g., [2, 11]). Note that the hybrid model is based on traffic flows, but is more accurate than a classical flow model since it handles discrete events such as packet drops and window adjusting.

3.1 Network Dynamics

Packet transmission rates are in the hybrid model treated as continuous-time real-valued variables. The received rate of packets at a router is denoted r and the sent rate is denoted s. The number of packets stored in a router queue is denoted q, which is also treated as a continuous variable. The dynamics of the queue is depending on if the queue is full ($q = q_{\max}$) or not ($0 \le q < q_{\max}$). We thus for each router introduce the hybrid system with two discrete states shown in Figure 2. In this model, subscript f refers to flow f, so that the windows w_f for all flows are updated according to given equations. Moreover, introduce the variables $q = \sum_f q_f$ and $s = \sum_f s_f$, and let the bandwidth of the outgoing link be equal to a constant B. Note that when the queue is full, a drop will be generated as soon as the variable z is equal to the predefined packet size L. Which flow f of the incoming flows that will lose a packet is determined by the distribution of the flows in the queue.

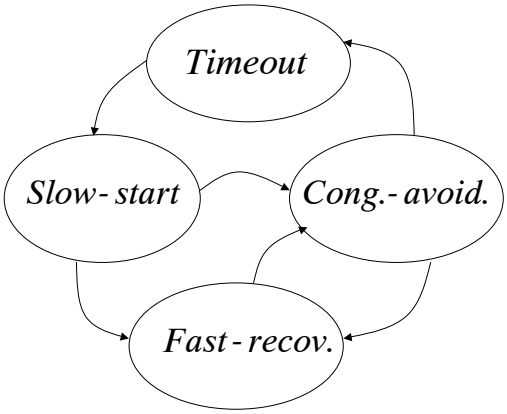

Figure 3: Hybrid TCP model.

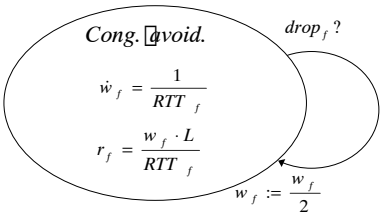

Figure 4: Simplified model of congestion control in TCP.

3.2 TCP Dynamics

The hybrid TCP model consists of four discrete states as shown in Figure 3. We will not detail the continuous dynamics and the transition rules for all states, but rather focus on the most important discrete state, namely, congestion avoidance. It is through the congestion avoidance state the additive increase and multiplicative decrease control strategy described in Section 2 is implemented.

Introduce the round-trip time RTT_f for flow f as the time between sending a packet and receiving the corresponding acknowledgment. It is given by the sum of the physical propagation time and the queueing times. The sender estimates RTT_f and uses it in the congestion control algorithm. Congestion avoidance in TCP can be described by the hybrid system in Figure 4. If RTT_f is approximately constant, we note that the window size w_f grows linearly. The corresponding transmission rate r_f for the sender is proportional to w_f. If a drop occur in flow f, the window size is reduced by a factor two.

Figure 5 shows a simulation of the queue size q (solid) and the window size w (dashed) for a typical session. When the queue becomes full ($q = q_{\max} = 57$), a packet is dropped and w is reduced by a factor two. The window size has a sawtooth shape, which is characteristic for TCP flows. The reason why the growth

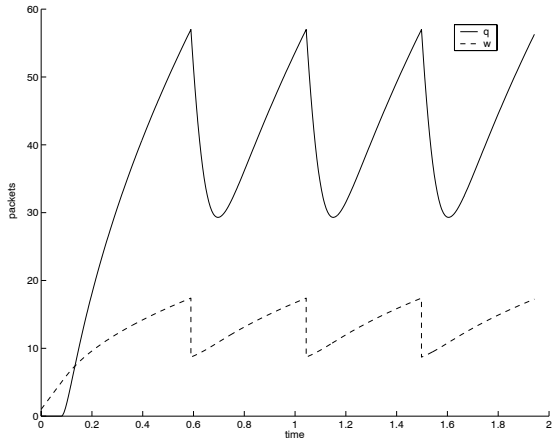

Figure 5: Illustration of congestion avoidance for a network with a single router. When the queue q (solid) exceeds $q_{\max} = 57$, one packets is dropped. TCP reduces accordingly the window w (dashed) by a factor two. Note the characteristic shape of w with intervals of approximately linear growth and periodic jumps.

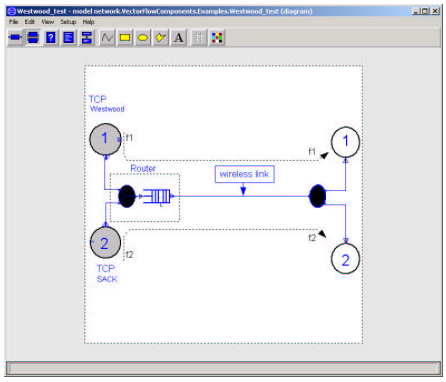

Figure 6: Example of a simple communication network implemented in Modelica.

in w is not linear in the beginning of the session is that RTT varies substantially due to the rapid increase in q: when q is small RTT is approximately equal to the physical propagation time, while if q is large packets spend a relatively large amount of time in the queue.

4 Modelica Implementation

An example of a communication network implemented in Modelica is shown in Figure 6. Two senders and two receivers are connected to the network. Their flows are sharing the same link capacity. If the sum of the flows at some time instance is larger than the bandwidth of the link, packets will be queued.

The hybrid model described in previous section consists of continuous equations and discrete events. Modelica was chosen as implementation language

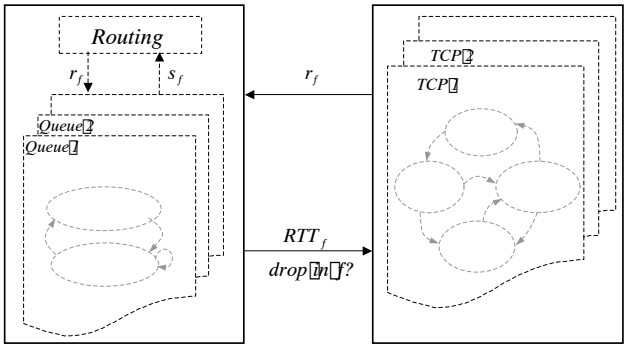

Figure 7: Composition of hybrid communication network model. The network dynamics and the TCP dynamics are separated.

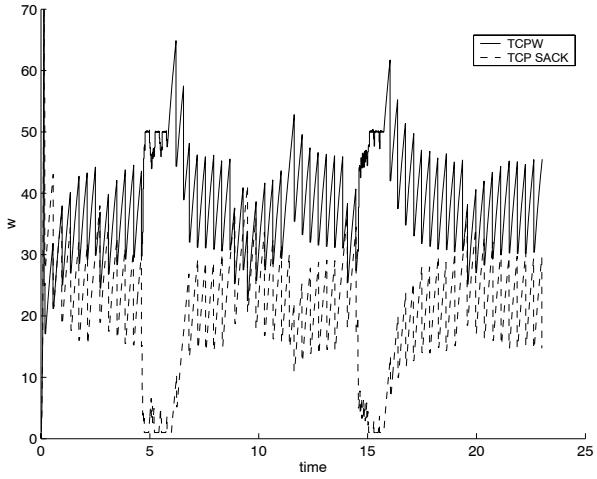

Figure 8: Simulation of the system in Figure 6. The windows size for TCP SACK is considerably reduced when the wireless link is losing a lot of packets (at about $t = 5$ and $t = 15$), while TCPW is able to cope with the losses very well.

since it supports efficient handling of such models. We have developed a communication network library. The library contains standard building blocks for network simulation, such as TCP senders, routing tables, and queues. Figure 7 shows the schematic layout of our communication network model. Note that the network dynamics and the TCP controllers are separated. The only information shared between the two submodels are the sending rates r_f, the round-trip times RTT_f, and the drop events. The modular structure allows an easy testing of for instance different TCP controllers applied to the same network topology.

Appropriate handling of the switching between discrete states is important for accurate and efficient simulation of the hybrid model. Implementations in Simulink showed some problems in this respect. Our implementation in Modelica and simulation in Dymola works well. Note that the simulation time is not depending on the number of flows, but instead by the number of discrete events generated by packet drops. The simulation time grows considerably when the number of discrete events becomes large, which hence limits the complexity of the model that can be studied. In `ns-2`, where individual packets are simulated, the simulation time is also depending on the size of the packet flows as well as the number of flows.

5 Example

Let us simulate the simple computer network in Figure 6. Sender 1 sends Flow f_1 using a version of TCP called TCP Westwood (TCPW) [6] and Sender 2 sends Flow f_2 using TCP SACK [3]. TCP SACK corresponds approximately to "standard" TCP used for the Internet today. The flows are sent over the same wireless link. For a wireless link transmission losses are more likely than for a wired link. For the network in Figure 6, there can be packet losses both due to that the router queue is full and due to that the wireless transmission lose packets. We model the wireless link as having a good and a bad state. In the good state 0.1% of the transmitted packets are lost, while in the bad state 10% of the packets are lost. The link stays in each state a random amount of time, which is exponentially distributed. TCPW was designed taking wireless links into account, while TCP SACK was designed for wired links. Next we will see that TCPW might be a good option for the emerging wireless Internet.

Figure 8 shows the window sizes for a simulation of the network in Figure 6. The solid line corresponds to TCPW and the dashed line to TCP SACK. Note the time intervals at about $t = 5$ and $t = 15$ when the link is in the bad state. The packet losses due to the bad transmission result in a sudden decrease of the window size for TCP SACK, while TCPW are able to compensate for the packet losses of the wireless link. The window size for TCPW is in general larger than for TCP SACK. This gives a larger throughput for the connection using TCPW, as is shown in Figure 9.

Figure 10 shows the throughput when two TCPW's are sharing the same wireless link (upper plots) and when two TCP SACK's are sharing the same link (lower plots). From the simulations we see that the major advantage of TCPW is when the link is in the bad state. When the link is in the good state, the performance of both TCP implementations are roughly equal.

Since TCP (SACK) was developed for wired net-

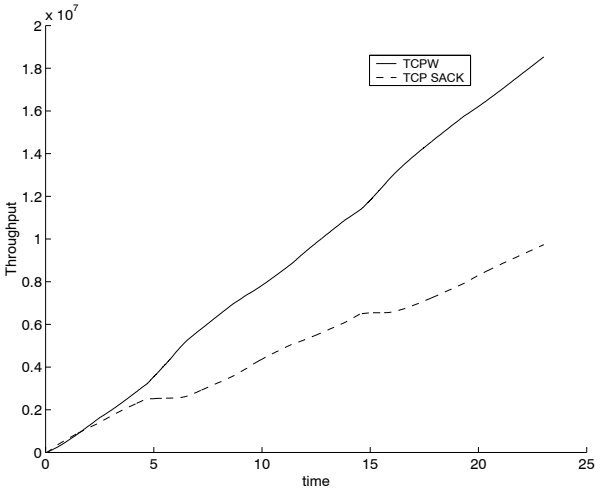

Figure 9: TCPW gives a larger throughput than TCP SACK. The difference is emphasized when there are a large number of packet losses due to the wireless transmission.

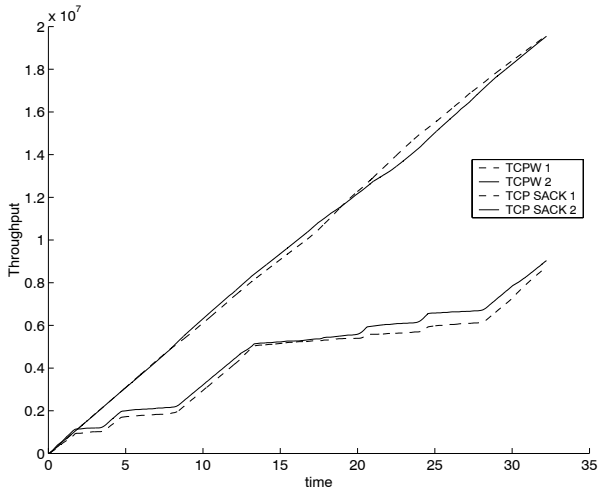

Figure 10: Throughput for two TCP Westwood senders (upper plots) and two TCP SACK senders (lower plots).

works, where packet losses arise only due to buffer overflow, there is a problem using it over wireless links. The reason is that the window size is reduced by a factor two every time a drop occurs, regardless if the drop is due to congestion or to transmission loss. TCPW has another way of updating the window size when a drop occurs. The window size is set to a value based on an estimate of the available bandwidth and the current round-trip time RTT. Since RTT is highly dependent on the queue sizes, so that RTT is small if and only if all corresponding queues are small, a small RTT implies that a detected drop must be due to wireless loss. The aim of TCPW is hence to utilize the available bandwidth more efficiently.

References

[1] S. Bohacek, J. P. Hespanha, J. Lee, and K. Obraczka. Analysis of a TCP hybrid model. In *Proc. of the 39th Annual Allerton Conference on Communication, Control, and Computing*, 2001.

[2] R. W. Brockett. Hybrid models for motion control systems. In H. Trentelman and J. Willems, editors, *Essays in Control: Perspectives in the Theory and Its Applications*, pages 29–53. Birkhäuser, Boston, 1993.

[3] 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.

[4] V. Jacobson. Congestion avoidance and control. In *Proc. of SIGCOMM*, volume 18.4, pages 314–329, 1988.

[5] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson. On the self-similar nature of ethernet traffic (extended version). *IEEE/ACM Transactions on Networking*, 2(1):1–15, 1994.

[6] S. Mascolo, C. Casetti, M. Gerla, M. Y. Sanadidi, and R. Wang. TCP Westwood: bandwidth estimation for enhanced transport over wireless links. In *MobiCom*, Rome, Italy, 2001.

[7] *The Network Simulator ns-2*. Information Sciences Institute, University of Southern California. `http://www.isi.edu/nsnam/ns`.

[8] V. Paxson and S. Floyd. Wide-area traffic: the failure of Poisson modeling. *IEEE/ACM Trans. on Networking*, 3(3):226–244, 1995.

[9] V. Paxson and S. Floyd. Why we don't know how to simulate the internet. In *Proc. of the Winter Simulation Conference*, 1997.

[10] L. L. Peterson and B. S. Davie. *Computer networks: a systems approach*. Morgan Kaufmann, 2nd edition, 2000.

[11] A. van der Schaft and J. M. Schumacher. *An Introduction to Hybrid Dynamical Systems*. Number 251 in Lecture Notes in Control and Information Sciences. Springer-Verlag, London, 2000.

[12] J. Walrand and P. Varaiya. *High-performance communication networks*. Morgan Kaufmann, 2nd edition, 2000.