HYBRID MODELING OF COMMUNICATION NETWORKS USING MODELICA

Daniel Färnqvist [*] **Katrin Strandemar** [*] **Karl Henrik Johansson** [*]
João Pedro Hespanha [**]

[*] *Department of Signals, Sensors & Systems*
Royal Institute of Technology, Stockholm, Sweden
`kallej@s3.kth.se`

[**] *Department of Electrical & Computer Engineering*
University of California, Santa Barbara, USA
`hespanha@ece.ucsb.edu`

Abstract: Modeling and simulation of communication networks using the modeling language Modelica are 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.

Keywords: Congestion control; Packet-switched networks; TCP; Internet; Modelica; Computer simulation; Hybrid systems

1. INTRODUCTION

The objective with research on control of networks 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 (Jacobson, 1988; Walrand and Varaiya, 2000; Peterson and Davie, 2000). 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 are far from satisfactory (Paxson and Floyd, 1995; Paxson and Floyd, 1997; Paganini *et al.*, 2001).

The intention of this paper is to describe initial work on modeling packet-switched communication network using Modelica (*Modelica*, n.d.; Mattsson *et al.*, 1998), which is an object-oriented language for modeling physical systems. Standard modeling and simulation environments targeted at network research include the discrete-event simulator ns-2 (*The Network Simulator ns-2*, n.d.). 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 packet transmission with a continuous flow and basically neglect the network protocols. A hybrid systems model, which is based on the average rates but takes packet drops and rate adjustments due to congestion control into account, was recently proposed in the literature (Hespanha *et al.*, 2001). 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 (Hespanha *et al.*, 2001; Bohacek *et al.*, 2001).

In this paper, we show that the hybrid model is suitable for Modelica. Moreover, we show that the model can be efficiently simulated using Dymola (Elmqvist *et al.*, n.d.), which is a commercial Modelica simulation environment.

The outline of the paper is as follows. In Section 2, a brief introduction to congestion control in communication networks is presented. 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. The paper ends in Section 6 with a brief discussion on ongoing work. An early version of this paper was presented as (Färnqvist *et al.*, 2002)

2. COMMUNICATION NETWORKS

A packet-switched network can be described by a graph, see Figure 1. The nodes represent the routers, which direct the packets from sender to receiver, and the edges correspond to wired or wireless links. 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 are leaving. 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 router 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).

In 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 its sending rate according to a control variable called the window size w. The TCP sender sends w number of packets and waits for their acknowledgments 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, the window size is increased by one: $w := w+1$. If a packet is dropped (so that no acknowledgment for that packet is received by the sender), 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. Moreover, using regular TCP over wireless links has shown to be inefficient, see further discussion in Section 5.

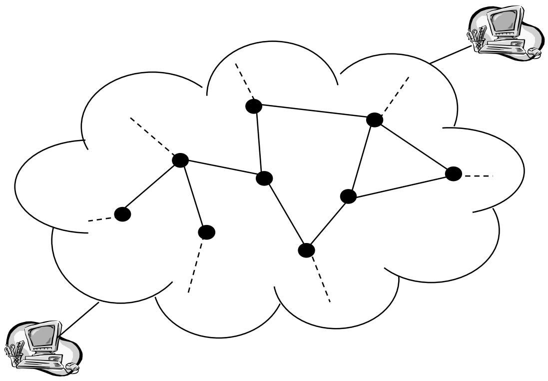

Fig. 1. Communication network. Packets from a sender to a receiver are transfered over wired or wireless links connected by routers. Since the links and the routers have limited capacity, congestion control strategies are needed.

3. HYBRID MODEL

In the hybrid model for communication networks proposed by Hespanha et al. (Hespanha *et al.*, 2001; Bohacek *et al.*, 2001), the network dynamics and the TCP dynamics are modeled as hybrid systems, e.g., (Witsenhausen, 1966; Brockett, 1993; van der Schaft and Schumacher, 2000; Johansson *et al.*, 2002). A hybrid system is a mathematical model for a dynamic system that has both continuous-time and discrete-event dynamics. The traffic dynamics of a packet-switched communication network is suitable to model as a hybrid system, where the packet flows are approximated by continuous dynamics while packet drops and control protocols are modeled as discrete changes. The motivation for introducing a hybrid model is that it captures the behavior of the system on a time scale in between conventional packet models and flow models. Thanks to the model abstraction of packets, the simulation time is in principle not affected by the number of packets transmitted or the queue sizes. Still, the hybrid model shows enough detail to accurately model transmission protocols and packet drops.

3.1 *Network Dynamics*

Packet transmission rates and queue lengths are modeled as continuous-time real-valued variables in the hybrid model. The arrival rate of packets at a router is denoted s and the departure rate is denoted r. The number of packets stored in a router queue is denoted q. The dynamics of the queue is depending on if the queue is full ($q = q_{\max}$) or not ($0 \leq q < q_{\max}$). For each router and flow, we introduce the hybrid system with two discrete states shown in Figure 2. In this model, subscript f refers to flow f, the variables q and s are defined as $q = \sum_f q_f$ and $s = \sum_f s_f$, and the bandwidth of the outgoing link is 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

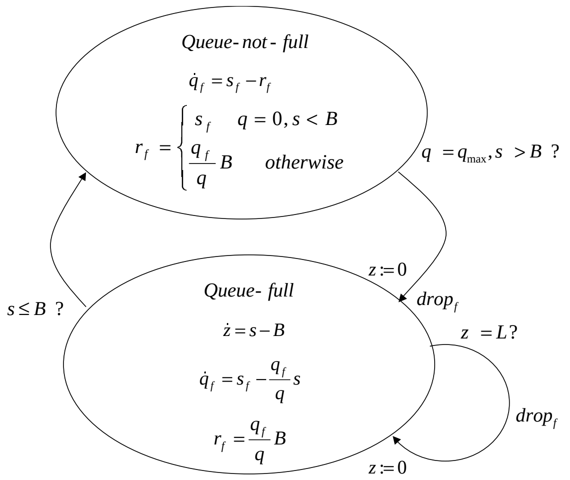

Fig. 2. Hybrid queue model. Each router is modeled by a number of two-state hybrid systems, each representing the queue dynamics for an individual flow f. As long as the queue is not full, the evolution of the queue size q_f is simply $\dot{q}_f = s_f - r_f$.

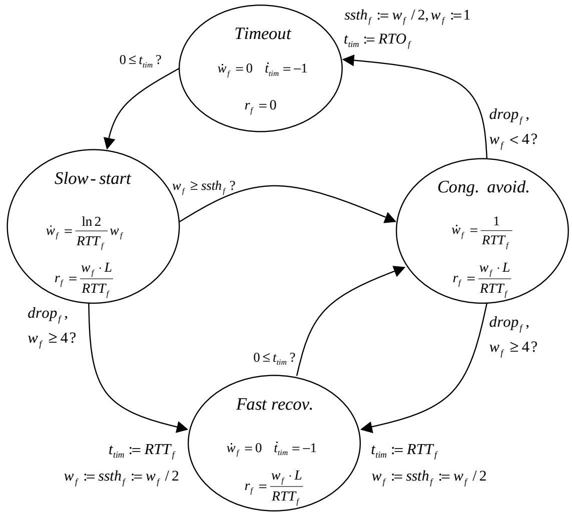

Fig. 3. Hybrid TCP model. Each TCP flow is modeled by a hybrid system with four discrete states. During a regular transmission session, the system is most of the time in the congestion avoidance state. In that state the window size grows approximately linearly: $\dot{w}_f = 1/RTT_f$.

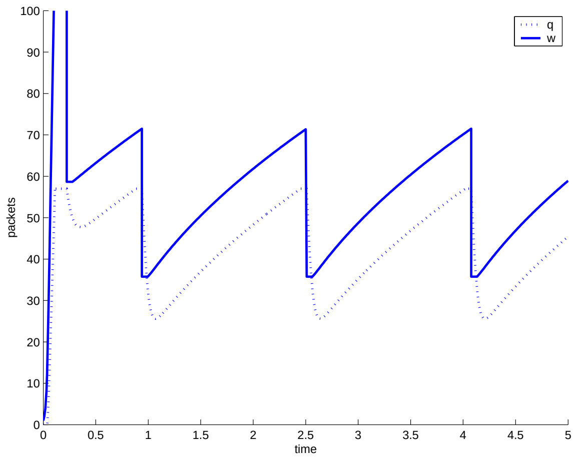

Fig. 4. Illustration of congestion avoidance for a network with a single router and a single flow. When the queue q (dashed) exceeds $q_{\max} = 57$, a packet is dropped. TCP reduces accordingly the window w (solid) by a factor two. Note the characteristic shape of w with intervals of approximately linear growth and periodic jumps.

lose a packet is determined by the distribution of the flows in the queue.

3.2 *TCP Dynamics*

For TCP, the sending rate r and the window size w are conveniently modeled as continuous variables, while the various modes of TCP (such as slow-start, congestion avoidance, and fast recovery) represent discrete states. The hybrid TCP model consists of four discrete states as shown in Figure 3. Here the round-trip time RTT_f for flow f is 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 hybrid TCP model uses RTT_f to update the window size w_f and to derive the sending rate r_f. These variables also depend on the discrete state of the TCP. Most of the time TCP is in the congestion avoidance state. Here w_f grows approximately linearly ($\dot{w}_f = 1/RTT_f$), since RTT_f is almost constant. When a drop occurs, w_f is reduced by a factor two.

3.3 *Simplest Example*

Let us illustrate TCP with the simplest example, namely, a single queue and a single flow. This system consists of one hybrid queue model of Figure 2 and one hybrid TCP model of Figure 3. Figure 4 shows a simulation of the queue size q (dashed) and the window size w (solid) for a typical TCP session. When the queue becomes full ($q = q_{\max} = 57$), a packet is dropped and w is reduced by a factor two. The evolution of the window size has a sawtooth shape (thanks to the AIMD control strategy) characteristic for TCP flows. Note the exponential increase of w in the beginning of the session. The reason for this is that TCP starts in the slow-start state, which (in spite of its name) gives a rapid increase of the sending rate (and accordingly, the window size) in the initial phase of a session.

4. MODELICA IMPLEMENTATION

Modelica is an object-oriented modeling language designed mainly for large heterogeneous physical systems. It is suitable for the hybrid network model described in previous section. For example, the continu-

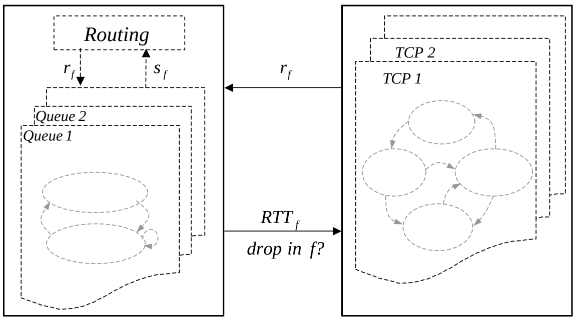

Fig. 5. Composition of hybrid communication network model. The network dynamics and the TCP dynamics are separated.

ous dynamics of the TCP model in Figure 3 is implemented through the following code:

```
if CongAvoid then
   der(w)=1/RTT;
   r=w*L/RTT;
elseif FastRecov then
   der(w)=0;
   r=w*L/RTT;
elseif TimeOut then
   der(w)=0;
   r=0;
elseif SlowStart then
   der(w)=log(2)*w/RTT;
   r=w*L/RTT;
end if;
```

Discrete events, such as the packet drops in the network model, can be implemented in the Modelica language as the following example:

```
when drop then
   reinit(w,w/2);
end when;
```

A communication network library was developed in Modelica. The library contains standard building blocks for network simulation, such as TCP senders, routing tables, and queues. Figure 5 shows the schematic layout of our communication network model. Note that the network dynamics and the TCP dynamics 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. Note that only the queue and the TCP components contain dynamical equations. The router component implement the network topology. The complexity of the model depends on the number of routers and the number of TCP flows. Note that each router–flow pair requires a hybrid queue model of Figure 2.

Appropriate handling of the switching between discrete states is important for accurate and efficient simulation of hybrid systems (Johansson *et al.*, 1999*a*; Johansson *et al.*, 1999*b*). 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 in general not depending on the queue sizes or sending rates, but instead depends on the number of discrete events generated by packet drops. The simulation time grows considerably when the number of discrete events becomes large. In ns-2, where individual packets are simulated, the simulation time depends on the size of the packet flows, the number of flows, and the queue sizes.

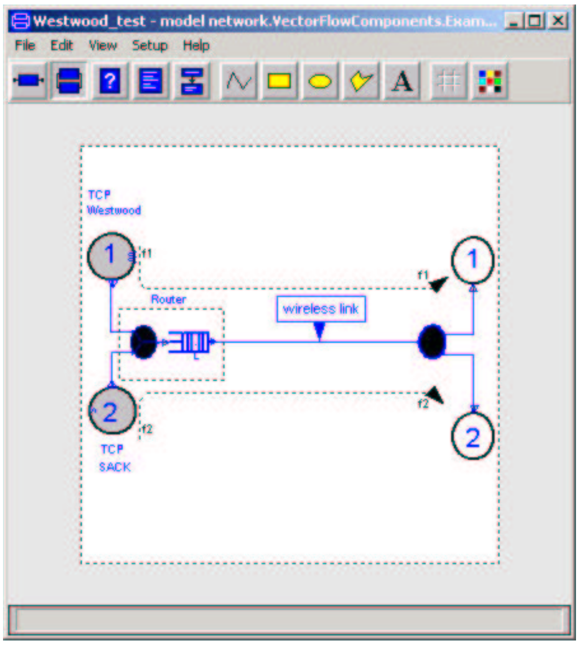

Fig. 6. Example of a communication network implemented in Modelica.

5. WIRELESS EXAMPLE

An example of a communication network implemented in Modelica is shown in Figure 6, where two senders and two receivers are connected to the network. Let us simulate this simple network. The two 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. Sender 1 sends Flow f_1 using a version of TCP called TCP Westwood (TCPW) (Mascolo *et al.*, 2001) and Sender 2 sends Flow f_2 using TCP SACK (Hespanha *et al.*, 2001). 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 either due to that the router queue is full or due to that the wireless transmission loses 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. This is illustrated next.

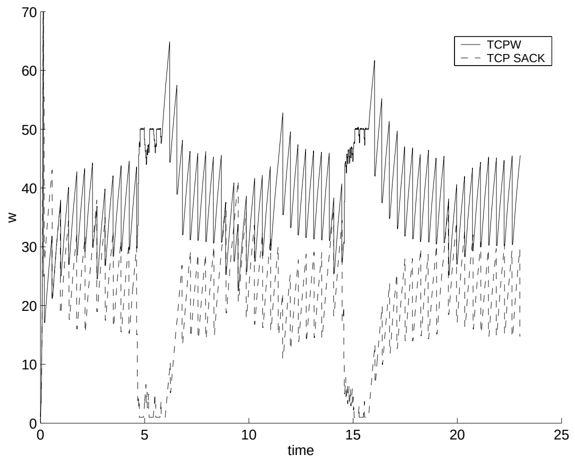

Fig. 7. Simulation of the system in Figure 6. The windows size for TCP SACK is considerably reduced when the wireless link is losing a lot of packets (at about $t = 5$ and $t = 15$), while TCPW is able to cope with the losses very well.

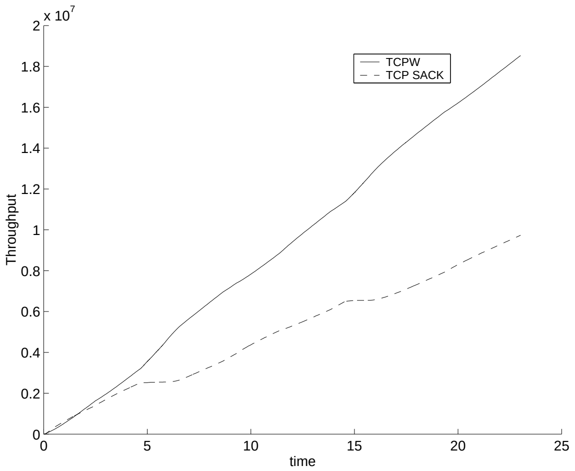

Fig. 8. TCPW gives a larger throughput than TCP SACK. The difference is emphasized when there are a large number of packet losses due to the wireless transmission.

Figure 7 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 8.

Figure 9 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.

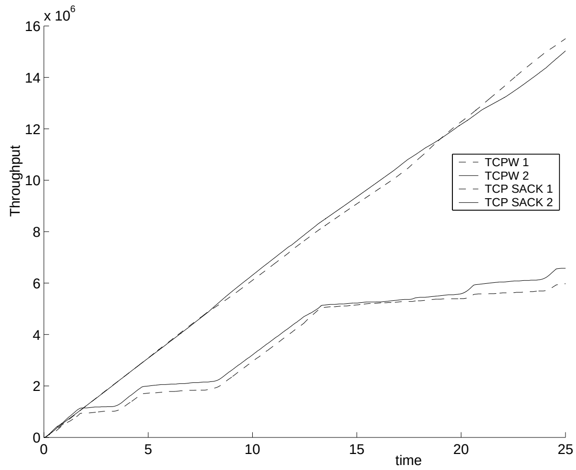

Fig. 9. Throughput for two TCP Westwood senders (upper plots) and two TCP SACK senders (lower plots).

Since TCP (SACK) was developed for wired networks, where packet losses arise mainly 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. Hence, the sending rate r will be reduced even if it is not needed. 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 only if all corresponding queues are small, a small RTT implies that a detected drop must be due to a wireless loss. The aim of TCPW is hence to utilize the available bandwidth more efficiently.

6. DISCUSSION

Heterogeneous networks with both wired and wireless links are important for many emerging applications. Ongoing work includes the comparison of various TCP variants, such as Reno, SACK, and Westwood, but also the proposal of improved congestion control mechanisms. An idea for router-controlled congestion avoidance is presented in (Auvert *et al.*, 2002), where so called network flaps are introduced to smoothing out bursty traffic flows.

The modeling environment that we have developed based on Modelica provides a flexible modular platform for simulating various networks and protocols. Note, however, that the hybrid model is not only suitable for efficient simulation, but also provides a mathematical framework suitable for analysis using tools from hybrid systems theory.

Acknowledgment

The authors want to thank Håkan Hjalmarsson and Gunnar Karlsson for helpful discussions. The work by Karl Henrik Johansson was supported by the Swedish Research Council.

7. REFERENCES

Auvert, M., H. Hjalmarsson, K. H. Johansson and G. Karlsson (2002). On router control for congestion avoidance. RVK.

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

Brockett, R. W. (1993). Hybrid models for motion control systems. In: *Essays in Control: Perspectives in the Theory and Its Applications* (H. Trentelman and J. Willems, Eds.). pp. 29–53. Birkhäuser, Boston.

Elmqvist, H., D. Brück and M. Otter (n.d.). *Dymola—User's Manual*. Dynasim AB. Research Park Ideon, Lund, Sweden.

Färnqvist, D., K. Strandemar, K. H. Johansson and J. P. Hespanha (2002). Hybrid modeling of communication networks using Modelica. In: *2nd International Modelica Conference*. Munich, Germany.

Hespanha, J. P., S. Bohacek, K. Obraczka and J. Lee (2001). Hybrid modeling of TCP congestion control. In: *Hybrid Systems: Computation and Control* (M. Di Benedetto and A. Sangiovanni-Vincentelli, Eds.). Vol. 2034 of *Lecture Notes in Computer Science*. pp. 291–304. Springer-Verlag. Berlin, Germany.

Jacobson, V. (1988). Congestion avoidance and control. In: *Proc. of SIGCOMM*. Vol. 18.4. pp. 314–329.

Johansson, K. H., J. Lygeros and S. Sastry (2002). Modeling of hybrid systems. In: *Encyclopedia of Life Support Systems*. UNESCO. Invited paper. Submitted.

Johansson, K. H., J. Lygeros, S. Sastry and M. Egerstedt (1999*a*). Simulation of Zeno hybrid automata. In: *Proc. 38th IEEE Conference on Decision and Control*. Phoenix, AZ.

Johansson, K. H., M. Egerstedt, J. Lygeros and S. Sastry (1999*b*). On the regularization of Zeno hybrid automata. *System & Control Letters* **38**, 141–150.

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

Mattsson, S. E., H. Elmqvist and M. Otter (1998). Physical system modeling with modelica. *Control Engineering Practice* **6**, 501–510.

Modelica (n.d.). `http://www.modelica.org`.

Paganini, F., J. Doyle and S. Low (2001). Scalable laws for stable network congestion control. In: *IEEE CDC*. Orlando, FL.

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

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

Peterson, L. L. and B. S. Davie (2000). *Computer networks: a systems approach*. 2nd ed.. Morgan Kaufmann.

The Network Simulator ns-2 (n.d.). Information Sciences Institute, University of Southern California. `http://www.isi.edu/nsnam/ns`.

van der Schaft, A. and J. M. Schumacher (2000). *An Introduction to Hybrid Dynamical Systems*. number 251 In: *Lecture Notes in Control and Information Sciences*. Springer-Verlag. London.

Walrand, J. and P. Varaiya (2000). *High-performance communication networks*. 2nd ed.. Morgan Kaufmann.

Witsenhausen, H. S. (1966). A class of hybrid-state continuous time dynamic systems. *IEEE Trans. Automatic Control* **11**(2), 161–167.