

# Fairness Analysis of TCP/IP

Eitan Altman, Chadi Barakat, Emmanuel Laborde  
INRIA, 2004 route des Lucioles, 06902 Sophia Antipolis, France  
E-mail: {Eitan.Altman,Chadi.Barakat, Emmanuel.Laborde}@sophia.inria.fr

Patrick Brown, Denis Collange  
France Telecom - CNET, 06921 Sophia Antipolis, France  
E-mail: {Patrick.Brown,Denis.Collange}@cnet.francetelecom.fr

## Abstract

Bandwidth sharing between multiple TCP connections has been studied under the assumption that the windows of the different connections vary in a synchronized manner. This synchronization is a main result of the deployment of Drop Tail buffers in network routers. The deployment of active queue management techniques such as RED will alleviate this problem of synchronization. We develop in this paper a mathematical model to study how the bottleneck bandwidth will be shared if TCP windows are not synchronized. This permits to evaluate the improvement in fairness and utilization brought by the deployment of active buffers. Also, this indicates how much a synchronization-based study underestimates the performance of TCP in a non-synchronized environment.

## 1 Introduction

One of the main objectives of TCP is to control the congestion in the Internet [11]. This control is not efficient if it does not ensure a fair sharing of network resources. A major problem of TCP is its bias against connections with long round-trip times (RTT) [3, 9, 12]. These connections are not able to achieve the same throughput as the other connections sharing the same path and having a smaller RTT. This is caused by the window increase algorithm adopted by TCP. Indeed, TCP uses an additive-increase multiplicative-decrease strategy for congestion control [11, 17]. It is known that such kind of strategies leads to fairness when all the connections increase their rates at the same rate [6]. We are talking here about a fairness in the sharing of the bandwidth of the bottleneck link regardless of the volume of resources consumed by a connection on the other links of the network. This kind of fairness is called in the literature the *max-min fairness* [9]. Other types of fairness however exist where the objective is to share fairly not only the resources at the bottleneck, but also the resources in other parts of the network. In case of TCP and in presence of connections of different RTT, a fairness cannot be ensured since the window increase rate

is inversely proportional to RTT (one packet per RTT in the congestion avoidance mode [17]) leading to an increase in the transmission rate at a rate inversely proportional to  $RTT^2$ . Note that the transmission rate of a window based protocol as TCP can be approximated at any moment by the window size divided by RTT. The connections with small RTT increase quickly their windows and grab most of the available bandwidth.

The throughput achieved by a TCP connection has been shown to be inversely proportional to  $T^\alpha$  with  $1 < \alpha < 2$  [12].  $T$  is the two-way propagation delay of the connection. This throughput has been calculated using the assumption that the windows of the different connections sharing the bottleneck vary in a synchronized manner [5, 12]. All the connections are supposed to reduce their windows simultaneously upon a period of congestion. This synchronization phenomenon has been indeed observed in the case of connections of close RTT [18]. It is mainly caused by the use of Drop Tail (DT) buffers. A DT buffer starts to drop arriving packets when it is filled thus informing the TCP sources of the occurrence of congestion. However, the reaction of the sources to these congestion signals takes one RTT to reach the congested buffer. During this time, the sources keep injecting packets into the network which results in losses from all the connections.

The synchronization phenomenon may disappear in future networks. Active queue management techniques such as RED (Random Early Detection) [10] have been recommended [4]. These new buffers aim to overcome the problems of DT ones. Congestion is anticipated and packets are dropped (or marked) before the overflow of the buffer. The drop is done early with a probability that increases with level of congestion. This avoids a large number of drops upon congestion. The probability that a flow loses a packet upon congestion is proportional to its share of the bandwidth. This early congestion detection together with the probabilistic drop has been shown to solve the problem of synchronization and thus to improve TCP fairness and bandwidth utilization [10]. With RED, a few number of congestion

signals is sent to flows consuming more than their fair share of the bandwidth. The other flows are protected from losses. Other buffer management techniques (e.g., FRED [14]) have been proposed to protect further flows using less than their fair share and to improve further TCP performance. Simple techniques as Drop from Front ones [13] have been also shown to be a solution for the problem of synchronization. In the case of these sophisticated buffers, a model based on the synchronization of flows is unable to evaluate correctly the performance of TCP congestion control.

Another problem with models assuming synchronization is that they use a fluid approach [5, 12] which does not account for the burstiness of TCP traffic. A fluid approach consists in supposing that TCP packets are spread over the path and not clustered in bursts. But, these fluid models deal with DT buffers which are known [10] to be unable to absorb the bursts of packets generated by TCP. TCP bursts may fill a DT buffer even before the full utilization of the bottleneck bandwidth. Fluid models assuming synchronization may then fail to evaluate correctly the performance of TCP even in a synchronized environment. An advantage of active buffers is that they are able to alleviate this bias against bursty traffic. By using the average length of the queue in congestion detection rather than the instantaneous length, and by fixing the threshold much lower than the total buffer size, the rapid fluctuations in queue length due to the arrival of bursts are absorbed [10]. Thus, fluid models should behave better in an environment where synchronization does not exist.

In this paper we develop a mathematical fluid model to study the performance of TCP when flows are not synchronized. This can be considered as a model for TCP in a network where active buffers such as RED are deployed. For simplicity of the analysis we consider the case of two concurrent connections. A generalization of the model to the case of multiple concurrent connections can be found in [2]. Instead of synchronization we assume that the connections reduce their windows upon congestion with a probability equal to their share of the bandwidth upon the congestion. A Markovian approach is used to solve this probabilistic model. We compare our numerical results to those of a model assuming synchronization, then we validate them via simulation. One of the results of our work is that approaches assuming synchronization lead to an underestimation of TCP performance. The absence of synchronization improves the fairness capacity of TCP. It can be considered as a mathematical result that shows the better fairness of TCP in case of active buffers.

In the next section we present our model. Section 3 contains the analysis. In Section 4 we compare the numerical results of our approach to those of an approach assuming synchronization. Simulation results are pre-

sented in Section 5. Section 6 concludes the work.

## 2 The mathematical model

Suppose that two TCP sources 1 and 2 share a path of bandwidth  $\mu$ . The two sources are assumed to have the same packet length. Denote the RTT of these connections by  $T_1$  and  $T_2$ . Denote also by  $W_1(t)$  and  $W_2(t)$  the window sizes of the two connections at time  $t$ . The rate of a connection at an instant  $t$  can be written as,

$$X_k(t) = W_k(t)/T_k \quad \text{with } k = 1, 2.$$

We assume that the two sources run a TCP version able to recover from losses without resorting to timeout and slow start. A SACK version or a New-Reno version can be used [8]. Upon loss detection, the TCP source divides its window by two, recovers from losses, and resumes then its window increase. We consider that the transfers are very long and we put ourselves in the stationary regime. We further assume that the queuing delay is small with respect to the propagation delay so that the RTT is approximately constant. This is reasonable with active buffers where the queue length is maintained at small values [10]. We consider the case when the window of TCP increases by one packet every RTT (i.e., delay ACK mechanism [17] is disabled; the analysis of the case of delay ACKs can be handled in exactly the same way our analysis below). The window and the rate of each source grow then linearly as a function of time as shown in [12] where a fluid model for the window evolution is used. We write for  $k = 1, 2$ ,

$$\frac{dW_k(t)}{dt} = \frac{dW_k(t)}{dack_k} \times \frac{dack_k}{dt} = \frac{1}{W_k(t)} \times \frac{W_k(t)}{T_k} = \frac{1}{T_k}.$$

This linear growth continues until a congestion occurs. Due to our assumption that queuing time is small, it is possible to consider that congestion occurs when the sum of the rates of the two connections reach the bottleneck bandwidth  $\mu$ . The difference in our model from previous models is the elimination of the synchronization hypothesis. A congestion event causes losses to one connection and only this connection divides its window by two. The window growth of the other connection is not affected. Given the probabilistic drop of packets at the onset of congestion, the probability that a specific connection is affected can be approximated by its share of the bottleneck bandwidth upon congestion.

**Definition 1** Denote by  $t_n$  the instant at which the  $n$ th congestion event occurs. Let  $W_1(t_n)$  (resp.  $W_2(t_n)$ ) be the window size of source 1 (resp. 2) just prior to this event. We assume that instants  $t_n$  are given by,

$$X_1(t_n) + X_2(t_n) = W_1(t_n)/T_1 + W_2(t_n)/T_2 = \mu. \quad (1)$$

We take the probability that a source  $k$  ( $k=1,2$ ) reduces its window at instant  $t_n$  equal to,

$$p_k = X_k(t_n)/\mu = W_k(t_n)/(\mu T_k).$$

We proceed now to the analysis of the performance of the two transfers. The aim is to calculate how fair they share the bottleneck bandwidth  $\mu$ , how well they utilize this bandwidth, and how much the network parameters affect the overall performance.

### 3 Throughput calculation

Given that the two processes  $W_1(t_n)$  and  $W_2(t_n)$  are related to each other by equation (1), we can transform the problem from a two-dimensional problem to a one dimensional-one. The study of one of the two processes is sufficient to describe the other. In the sequel we focus on the study of connection 1. We start by calculating the relationship between  $W_1(t_n)$  and  $W_1(t_{n+1})$  as well as the time  $(t_{n+1} - t_n)$  between two consecutive congestion events. The window variation as a function of time and the sum of the rates at instants  $t_n$  and  $t_{n+1}$  are used. First we state the main results,

**Theorem 1** *If connection 1 is hurt by congestion at instant  $t_n$ , the next congestion will appear after a time,*

$$t_{n+1} - t_n = \frac{T_1 T_2^2}{T_1^2 + T_2^2} \times \frac{W_1(t_n)}{2},$$

*and the window size of connection 1 prior to this next congestion event will be equal to,*

$$W_1(t_{n+1}) = \frac{T_1^2 + 2T_2^2}{2(T_1^2 + T_2^2)} \times W_1(t_n). \quad (2)$$

*If connection 2 is hurt by the congestion, connection 1 continues to increase its window without reduction until the next congestion event which occurs after a time,*

$$t_{n+1} - t_n = \frac{T_1^2 T_2^2}{2(T_1^2 + T_2^2)} \times \left( \mu - \frac{W_1(t_n)}{T_1} \right).$$

*In this case, the window of connection 1 prior to the next congestion event will be equal to,*

$$W_1(t_{n+1}) = \frac{T_1 T_2^2}{2(T_1^2 + T_2^2)} \times \left( \mu - \frac{W_1(t_n)}{T_1} \right) + W_1(t_n). \quad (3)$$

**Proof:** Suppose first that it is connection 1 which suffers from losses at time  $t_n$ . It divides its window by two while source 2 continues to increase its window without reduction. At time  $t_{n+1}$  we can write,

$$W_1(t_{n+1}) = W_1(t_n)/2 + (t_{n+1} - t_n)/T_1$$

$$W_2(t_{n+1}) = W_2(t_n) + (t_{n+1} - t_n)/T_2$$

$$W_2(t_{n+1}) = T_2 (\mu - W_1(t_{n+1})/T_1)$$

$$W_2(t_n) = T_2 (\mu - W_1(t_n)/T_1)$$

The solution of this system of equations in  $W_1(t_{n+1})$  and  $(t_{n+1} - t_n)$  as a function of  $W_1(t_n)$  concludes the

first part of the proof. The second part corresponds to the case when connection 2 is hurt by losses. At time  $t_{n+1}$  we can write,

$$W_1(t_{n+1}) = W_1(t_n) + (t_{n+1} - t_n)/T_1$$

$$W_2(t_{n+1}) = W_2(t_n)/2 + (t_{n+1} - t_n)/T_2$$

$$W_2(t_{n+1}) = T_2 (\mu - W_1(t_{n+1})/T_1)$$

$$W_2(t_n) = T_2 (\mu - W_1(t_n)/T_1)$$

The solution of this system concludes the proof.  $\blacksquare$

The state distribution of connection 1 at instant  $t_{n+1}$  is only a function of its state at time  $t_n$ . Thus, the stochastic process  $W_1(t_n)$  forms a Markov process. In order to calculate numerically its stationary distribution, we discretized the space of this process and we described it with a Markov chain. Let  $\mathcal{I} \subset [1, W_1^{MAX}]$  denote the state space of this chain and let  $P = (p_{ij})_{i,j \in \mathcal{I}}$  denote its transition matrix. The window of the connection is assumed to have a minimum size of 1 packet and a maximum size of  $W_1^{MAX}$  packets.  $W_1^{MAX}$  is calculated using equation (1) when the other connection is at a window of 1 packet. Using Theorem 1 as well as the probability that a connection suffers from congestion at instant  $t_n$ , we can find the transition matrix  $P$ .

Suppose that connection 1 is in state  $i$  at time  $t_n$ . Define  $g(i)$  as the state of this connection at time  $t_{n+1}$  when it is hurt by the congestion at time  $t_n$ .  $\hat{g}(i)$  denotes its state if connection 2 is hurt by the congestion at time  $t_n$ . Thus, using equations (2) and (3),

$$g(i) = \frac{T_1^2 + 2T_2^2}{2(T_1^2 + T_2^2)} \times i \quad (4)$$

$$\hat{g}(i) = \frac{T_1 T_2^2}{2(T_1^2 + T_2^2)} \times \left( \mu - \frac{i}{T_1} \right) + i \quad (5)$$

Matrix  $P = (p_{ij})_{i,j \in \mathcal{I}}$  can then be written as,

$$p_{ij} = \begin{cases} p_1 = \frac{i}{\mu T_1} & \text{if } j = g(i) \\ p_2 = 1 - p_1 = 1 - \frac{i}{\mu T_1} & \text{if } j = \hat{g}(i) \\ 0 & \text{otherwise} \end{cases}$$

In almost all cases, the Markov chain  $\{W_1(t_n)\}$  turned out to be irreducible. To check this, we used techniques that calculate the transitive closure of a graph [7]. When the Markov chain is irreducible, it has a unique stationary regime. Let  $\pi = (\pi_i)_{i \in \mathcal{I}}$  denote its stationary distribution. In the following sections we show how to calculate the throughput from the stationary regime of this Markov chain. We define for this purpose a semi-Markov process and some cost functions.

#### 3.1 Definition of a semi-Markov process

Let us define the process  $A(t)$  as being equal to,

$$A(t) = W_1(t_n) \quad \text{for } t_n \leq t < t_{n+1}.$$

The transition time of this process depends on the current and the next state, and this time is not exponentially distributed.  $A(t)$  forms then a semi-Markov process. Its average over a long time interval is different from the average of process  $W_1(t)$  given that  $W_1(t)$  varies linearly between two jumps of  $A(t)$ . To eliminate this discrepancy between  $A(t)$  and  $W_1(t)$ , we defined some cost functions that depend on the current state of process  $A(t)$ . We weighted then the integral of process  $A(t)$  over a long time interval with these cost functions in order to transform it to the integral of process  $W_1(t)$ .

### 3.2 Definition of cost functions

Suppose that  $A(t)$  visits state  $i$  then jumps to state  $j$  on the next congestion event. We define  $f_{ij}$  as the integral of  $W_1(t)$  between these two transitions. We denote the time between these transitions by  $\tau_{ij}$ . Our cost function associated to state  $i$  is defined as the expected value of  $f_{ij}$  over all the possible values of  $j$ . We denote this cost function by  $f_i$ . Using (4) and (5), it follows that

$$\begin{aligned} f_i &= \sum_{j \in \mathcal{I}} f_{ij} p_{ij} = f_{ig(i)} p_{ig(i)} + f_{i\hat{g}(i)} p_{i\hat{g}(i)} \\ &= i/(T_1 \mu) \int_0^{\tau_{ig(i)}} (t/T_1 + i/2) dt \\ &\quad + (1 - i/(T_1 \mu)) \int_0^{\tau_{i\hat{g}(i)}} (t/T_1 + i) dt \end{aligned}$$

where  $\tau_{ig(i)}$  and  $\tau_{i\hat{g}(i)}$  are given by Theorem 1. We denote by  $\tau_i$  the average time  $A(t)$  stays in state  $i$ ,

$$\tau_i = \sum_{j \in \mathcal{I}} \tau_{ij} p_{ij} = \tau_{ig(i)} p_{ig(i)} + \tau_{i\hat{g}(i)} p_{i\hat{g}(i)}.$$

### 3.3 Calculation of the throughput

The throughput of connection 1 is equal to the time average of its congestion window divided by  $T_1$

$$\overline{X}_1 = \lim_{t \rightarrow \infty} \frac{1}{t} \int_0^t \frac{W_1(\tau)}{T_1} d\tau$$

Using the theory of Markov reward processes (or of delayed regenerative processes) [15], it can be proved that this limit exists and is equal to

$$\overline{X}_1 = \frac{\sum_{i \in \mathcal{I}} \pi_i f_i}{\sum_{i \in \mathcal{I}} \pi_i \tau_i}, \quad P - a. s.$$

Now for connection 2, the relationship between  $\{W_1(t_n)\}$  and  $\{W_2(t_n)\}$  is used to avoid the repetition of all the work. To every state of the Markov chain associated to connection 1 corresponds a state of the Markov chain associated to connection 2. Only the cost functions for connection 2 need to be recalculated. The stationary distribution as well as the average time between congestion events don't change. The throughput of connection 2 is again calculated by dividing the time average of  $W_2(t)$  by  $T_2$ .

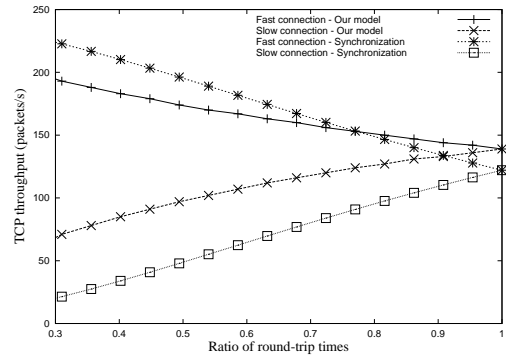


Figure 1: Comparison of throughputs

## 4 Numerical results

We solved numerically our model for the throughputs of the two connections. We compare then our results to those of a model assuming synchronization between flows. We chose the model in [12] and we simplified it to match our hypothesis on the queuing time.

### 4.1 The synchronization case

In [12] the authors suppose that in the stationary regime, the window of connection  $k$  ( $k = 1, 2$ ) varies in a linear and periodic manner between two values  $w_k$  and  $2w_k$ . They approximate the throughput of a connection by,  $\overline{X}_k = 3w_k/2T_k$ . Using the synchronization assumption, they showed that the window of a connection is inversely proportional to its RTT. The sum of the rates of the two connections upon congestion is taken equal to the bottleneck bandwidth. They wrote

$$w_1/w_2 = T_2/T_1, \quad 2w_1/T_1 + 2w_2/T_2 = \mu,$$

which yields

$$\overline{X}_1 = \frac{3}{4} \frac{T_2^2}{T_1^2 + T_2^2} \mu, \quad \overline{X}_2 = \frac{3}{4} \frac{T_1^2}{T_1^2 + T_2^2} \mu.$$

### 4.2 Comparison of numerical results

In Figure 1, we plot the throughputs of the two connections as a function of the ratio of their RTT. We mean by slow connection the one with the long RTT. The other connection is called the fast connection. Four lines are plotted, two for our model and two for the synchronization case. The bottleneck bandwidth is taken equal to 1.5 Mbps together with TCP packets of total size 576 bytes. The RTT of the slow connection is fixed to 0.5 s. That of the fast one is varied. The X-axis represents the ratio of the small RTT and the long one.

We notice first that the throughput achieved by the slow connection is better in our case. Given that it has a small throughput, there is a small probability that the slow connection reduces its window upon congestion. This gives it better performance. However, when the two connections are synchronized, the slow connection is obliged to reduce its window with the fast one.

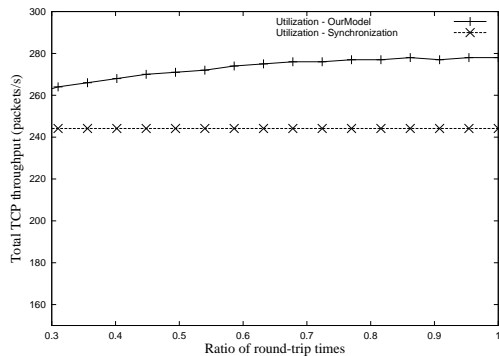


Figure 2: Comparison of utilizations

This is equivalent to a drop probability equal to one instead of the bandwidth share. Given that it increases its window much slower, the slow connection gets poorer performance when we force it to reduce always its window.

The increase in the performance of the slow connection in our case is accompanied by a decrease in the performance of the fast one. However, the deterioration in the performance of the fast connection is not as important as the improvement in the performance of the slow one. This means that our model predicts better utilization of the bottleneck bandwidth than the model in [12]. This is illustrated in Figure 2. Indeed, when a congestion occurs, the sum of the rates is equal to  $\mu$ . In our case, one of the two connections reduces its window and then the reduction in the total rate is less than half  $\mu$ . However in the synchronization case, the two connections divide their windows simultaneously and the reduction in the total rate is equal to  $\mu/2$ . Thus, in our case, the utilization is kept at higher levels.

The synchronization model predicts that the throughput of a connection is inversely proportional to  $RTT^2$ . One must predict that our model will give something less than that. The ideal case is when the performance of a connection is independent of RTT. We plot in Figure 3 the ratio of the throughput of the fast connection and that of the slow connection. These are the throughputs shown in Figure 1. We plot on the same figure different powers of  $T_2/T_1$  to see to which power our model is close. The X-axis is that of  $T_1/T_2$ . The numerical results of our model are located between the line of power 0.8 and that of power 0.9. The throughput of a connection can be then supposed to be inversely proportional to  $(RTT)^{0.85}$  in case flows are not synchronized instead of  $RTT^2$  in the synchronization case.

## 5 Simulation

### 5.1 Simulation scenario

We simulate two long TCP transfers over a bottleneck node using the `ns-2` simulator [16]. The version TCP-SACK [8] is used. The simulation scenario is shown

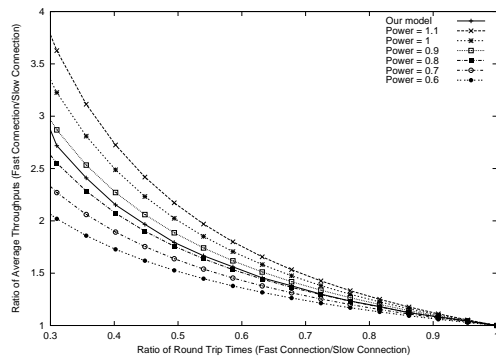


Figure 3: The ratio of throughputs

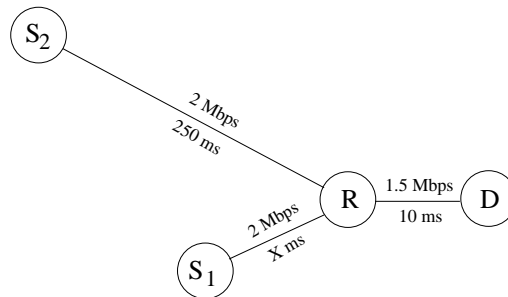


Figure 4: The simulation scenario

in Figure 4. Two sources  $S_1$  and  $S_2$  are connected to a router  $R$  in order to reach the destination  $D$ . We vary the propagation delay of the link between  $S_1$  and  $R$  between 40 ms and 200 ms. Simulations are run for 500s each. The receiver windows are set large enough so that the window is only limited by network parameters.

### 5.2 Case of a RED buffer

We supply router  $R$  with a RED buffer of a total size of 20 packets, a minimum threshold of 5 packets and a maximum threshold of 10 packets. The maximum drop probability is taken equal to 0.1 and the weight used in the calculation of the average queue size is taken equal to 0.002. The reason for taking small thresholds is to minimize the queuing time.

In Figure 5 we plot the throughput of every connection as a function of the propagation delay between  $S_1$  and  $R$ . The results are closer to those of our model especially for the slow connection. This connection gets more bandwidth than what is predicted in the synchronization case. The RED buffer with its probabilistic drop alleviates the problem of synchronization and improves the fairness of TCP. This improvement is better observed for small RTT of connection 1. When the RTT of this connection increases, its reaction to packet losses becomes slower. The reaction of the slow connection is already slow. Then, it becomes more likely that upon a congestion period the RED buffer drops packets from both connections. The two connections start to reduce their windows simultaneously which is not considered by our model. Here, the simulation results get closer to

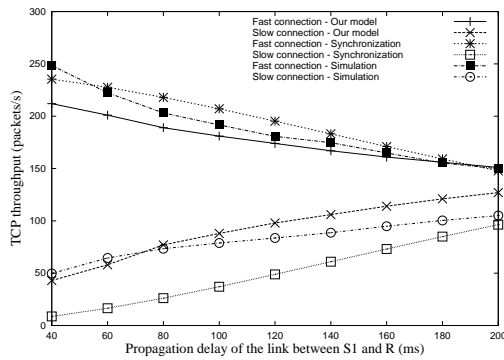


Figure 5: Throughputs in case of RED buffer

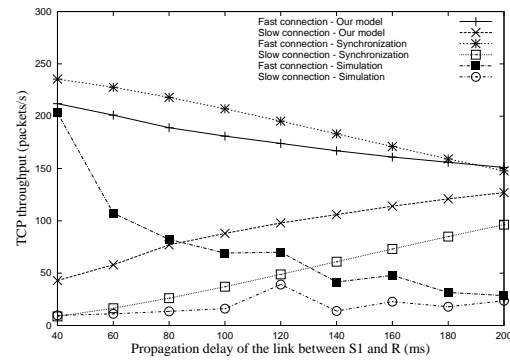


Figure 6: Throughputs in case of DT buffer

those of a model assuming synchronization.

### 5.3 Case of a Drop Tail buffer

We repeat our simulation with a small DT buffer in router  $R$ . We set the buffer size to the minimum threshold of the RED buffer in the previous simulation. Results are plotted in Figure 6. The bias of DT buffers against bursty traffic leads to a poor performance for both connections. The fluid models don't work correctly in this case. The congestion appears when two bursts arrive simultaneously at the buffer and not when the bandwidth is fully utilized. A model assuming synchronization can be used in this case but the instants of congestion cannot be predicted. All that we can say with a fluid model is that the windows of the two connections are inversely proportional to their RTT.

## 6 Conclusions

We proposed a Markovian fluid model to study the fairness of TCP when connections are not synchronized. The absence of synchronization is claimed to be one of the main results of active queue management techniques such as RED. We showed that the fairness of TCP improves in a non-synchronized environment. We showed also that the absence of synchronization improves the utilization of network resources. We validated these results with simulations. The burst absorption capacity of active buffers improves the accuracy of fluid models for TCP. Drop Tail buffers are biased against bursty traffic and fluid models don't work well especially in case of small buffers.

### References

[1] E. Altman, J. Bolot, P. Nain, D. Elouadghiri- M. Er-randani, P. Brown, and D. Collange, "Performance Modeling of TCP/IP in a Wide-Area Network", *IEEE Conference on Decision and Control*, Dec 1995.

[2] C. Barakat and E. Altman, "A Markovian Model for TCP Analysis in a Differentiated Services Network", *Workshop on Quality of future Internet Services*, Sep 2000.

[3] C. Barakat, E. Altman, and W. Dabbous, "On TCP Performance in a Heterogenous Network: A Survey", *IEEE Communications Magazine*, Jan 2000.

[4] B. Braden, et al., "Recommendations on Queue Management and Congestion Avoidance in the Internet", *RFC 2309*, Apr 1998.

[5] P. Brown, "Resource sharing of TCP connections with different round trip times", *IEEE Infocom*, Mar 2000.

[6] D. Chiu and R. Jain, "Analysis of the Increase/Decrease Algorithms for Congestion Avoidance in Computer Networks", *Journal of Computer Networks and ISDN*, Jun 1989.

[7] T. Cormen, C. Leiserson, and R. Rivest, "Introduction to Algorithms", *The MIT Press*, Cambridge, Massachusetts.

[8] K. Fall and S. Floyd, "Simulation-based Comparisons of Tahoe, Reno, and SACK TCP", *Computer Communication Review*, Jul 1996.

[9] S. Floyd, "Connections with Multiple Congested Gateways in Packet-Switched Networks Part 1: One-way Traffic", *Computer Communication Review*, Oct 1991.

[10] S. Floyd and V. Jacobson, "Random Early Detection gateways for Congestion Avoidance", *IEEE/ACM Transactions on Networking*, Aug 1993.

[11] V. Jacobson, "Congestion avoidance and control", *ACM SIGCOMM*, Aug 1988.

[12] T.V. Lakshman and U. Madhow, "The performance of TCP/IP for networks with high bandwidth-delay products and random loss", *IEEE/ACM Transactions on Networking*, Jun 1997.

[13] T.V. Lakshman, A. Neidhardt, and T.J. Ott, "The Drop from Front Strategy in TCP over ATM and its Interworking with other Control Features", *IEEE INFOCOM*, 1996.

[14] D. Lin and R. Morris, "Dynamics of Random Early Detection", *ACM SIGCOMM*, Sep 1997.

[15] S. M. Ross, "Applied Probability Models with Optimization Applications", *Holden-Day*, San Francisco, 1970.

[16] Network Simulator v.2, University of California at Berkeley, Available via <http://www.nrg.ee.lbl.gov/ns-2>.

[17] W. Stevens, "TCP Slow-Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms", *RFC 2001*, Jan 1997.

[18] L. Zhang, S. Shenker, and D.D. Clark, "Observations on the Dynamics of a Congestion Control Algorithm: The Effects of Two-Way Traffic", *ACM SIGCOMM*, Sep 1991.