Analysis of Scalable TCP

Ralph El Khoury and Eitan Altman
INRIA, BP93, 06902 Sophia Antipolis, France

Abstract

In recent years, several more aggressive versions of TCP have been proposed which leave the additive increase multiplicative decrease (AIMD) paradigm. The motivation is to adapt TCP to networks with very large bandwidth delay products. In this paper we study the scalable TCP (STCP) which is designed to be a Multiplicative Increase Multiplicative Decrease (MIMD) protocol. We analyze how it shares a bottleneck link with other CBR flows. We identify conditions under which the time varying queueing delay transforms the whole multiplicative increase dynamics of the window size into an additive increase dynamics, which implies that window evolution of STCP behaves as an AIMD protocol.

1 Introduction

In high speed networks, the congestion avoidance phase of TCP takes a long time to increase the window size and fully utilize the available bandwidth. S. Floyd writes in [2]: "for a Standard TCP connection with 1500-byte packets and a 100 ms round-trip time, achieving a steady-state throughput of 10 Gbps would require an average congestion window of 83,333 segments, and a packet drop rate of at most one congestion event every 5,000,000,000 packets (or equivalently, at most one congestion event every $1\frac{2}{3}$ hours).The average packet drop rate of at most 2×10^{-10} needed for full link utilization in this environment corresponds to a bit error rate of at most 2×10^{-14}, and this is an unrealistic requirement for current networks." This has motivated in recent year the design of new versions of TCP for networks with high bandwidth delay product: the HighSpeed TCP [2] as well as MIMD versions of TCP (Scalable TCP – STCP [5], High-Throughput TCP [8]), the Binary Increase Congestion Control [7] and FAST TCP [4].

In this paper we analyze the behavior of STCP that shares a bottleneck link with CBR non-responsive traffic. We present a fluid approximation model where the discrete values of the window size are replaced by a real valued variable. We obtain the average throughput of STCP and compare it to that obtained by an AIMD congestion control. We identify two phases in the behavior of the instantaneous window size: a part of exponential increase in time that corresponds to the period in which the buffer is empty, and a linear increase during the part in which the buffer is non-empty. (The occurrence of two phases arises also in other versions of TCP, see [1] who study a similar setting but with the Tahoe version). We identify conditions under which only the second phase exists, which means that the time evolution of the window size of STCP follows an AIMD dynamics. Our model is finally validated through simulations and the parameters for which the model is valid are identified. Modifications are proposed for some extreme cases.

2 System model

We consider a shared bottleneck link depicted in Fig 1, where two streams share the bottleneck. The first is STCP, and the second one is a deterministic exogenous traffic with a constant rate (later in the simulations we will consider a UDP stream). The exogenous traffic is independent of the load of the network. The parameters used for the system are:

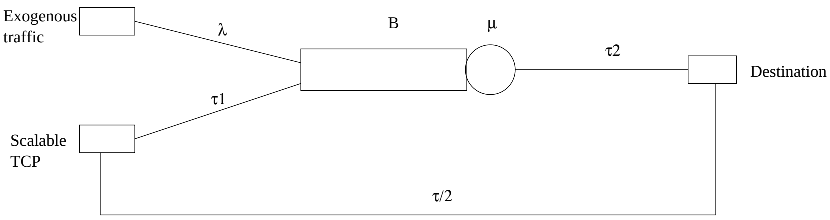

Figure 1: System model

μ: service rate of the queue.
λ: the rate of the exogenous traffic.
τ_1: the delay between the STCP source and the buffer.
τ_2: the delay between the buffer and the destination.
$\tau \triangleq 2(\tau_1 + \tau_2)$
$T \triangleq \tau + 1/\mu$
B: the buffer size.
$Q(t)$: instantaneous size of the buffer.
$W(t)$: window size at time t.
W_{loss}: window size at the end of a congestion avoidance phase before a loss occurs.
W_{max}: maximal size of $W(t)$ at the instant of the loss detection
a: the increase parameter of the window.
$\beta \triangleq 1 - b$: the decrease parameter of the window.
thp_{in}: throughput of the controlled source between the source and the buffer.
thp_{out}: throughput of the controlled source between the buffer and the destination.

In examples, when units are not given, rates will be in Mbps and time in msec.

3 Computation of the average throughput of STCP

Motivated by simulations and by analysis of other versions of TCP [1], we consider a periodic behavior of STCP (where a cycle corresponds to the time between consecutive loss events) We thus study the behavior of one cycle, and from there we derive the expression of the throughput. We have considered in our examples, as recommended in [5], an increase factor $a = 0.01$, and a decrease factor $b = 0.875$. We will use a fluid model for the window size $W(t)$ at time t to study the performance of STCP.

For each acknowledgement (ACK) received in a round trip time in which congestion has not been detected, we have: $W(t^+) = W(t) + a \; \forall \; W(t)$. The rate of the window growth with arriving ACKs is: $\dfrac{dW(t)}{dack} = a, \; \forall \; W(t)$. The ACKs are generated at a rate equal to thp_{out}. Hence

$$\frac{dW(t)}{dt} = \frac{dW}{dack}\frac{dack}{dt} = \frac{dW}{dack} thp_{out}.$$

We see that the window grows at a rate equal to:

$$\frac{dW(t)}{dt} = a \cdot thp_{out} \forall \; W(t) \tag{1}$$

We distinguish two phases:
(i) When the queue is empty: $thp_{out}(t) = thp_{in}(t) = W(t)/T$
(ii)When the queue builds: thp_{in} is larger than from thp_{out}. We assume like in [1] that the output rate of the STCP is proportional to its input rate: $thp_{out}(t) = \mu thp_{in}(t)/(thp_{in}(t) + \lambda)$. Then,

$$thp_{in}(t) = thp_{out}(t).\frac{\lambda}{\mu - thp_{out}(t)} \tag{2}$$

and we know that:

$$thp_{in}(t) = \frac{dW}{dt} + \frac{dack}{dt} = (1 + \frac{dW}{dack})thp_{out}(t). \tag{3}$$

We obtain from the two last equations,

$$thp_{out}(t) = \mu - \frac{\lambda}{(1 + \frac{dW}{dack})} \tag{4}$$

We set $\hat{T}$ the time at which the queue starts to build up, and $\hat{W}$ the corresponding window size.

$$thp_{out} = \begin{cases} \frac{W(t)}{T} & \text{if } W < \hat{W} \\ \mu - \frac{\lambda}{1+a} & \text{if } W > \hat{W} \end{cases} \tag{5}$$

Therefore,

$$\frac{dW(t)}{dt} = \begin{cases} a\frac{W}{T} & \text{if } W < \hat{W} \\ a(\mu - \frac{\lambda}{1+a}) & \text{if } W > \hat{W} \end{cases} \tag{6}$$

It appears from our last equation that the window grows in two different ways in a one cycle for $W < \hat{W}$, and linearly for $W > \hat{W}$. These two behaviors depend on the value of W.

Let us now derive the expression of W_{loss}. When the queue is full, it contains $B \cdot (thp_{in})/(thp_{in} + \lambda)$ STCP packets. Already, we know the throughputs in and out corresponding to this case. Therefore, we obtain as in [1]

$$W_{loss} = \tau_1 \cdot thp_{in} + (\tau_2 + \tau/2).thp_{out} + B \cdot \frac{thp_{in}}{thp_{in} + \lambda} + 1$$

(Note: the 1 in the equation is due to the packet currently in service). We set, $W_{max} = W_{loss} + a \cdot W_{loss} = (1+a)W_{loss}$. We have $W_{max} > \hat{W}$. Then $thp_{out} = \mu - \lambda/(1+a)$ and from eq (2)

$$thp_{in}(t) = thp_{out}(t).\frac{\lambda}{\mu - thp_{out}(t)} = (1+a)(\mu - \lambda/(1+a))$$

which yields

$$W_{max} = (1+a)(\mu - \lambda/(1+a))((2+a)\tau_1 + 2\tau_2 + B/\mu) + 1. \tag{7}$$

We can also write it by using $T = \tau + 1/\mu$:

$$W_{max} = (1+a)(\mu - \lambda/(1+a))(1+a)\tau_1 + T + B/\mu) + \lambda/((1+a)\mu) \tag{8}$$

Define
T_1 and T_2: durations of the first and the second phase respectively of a cycle.
N_1 and N_2: number of packets sent in T_1 and T_2 respectively.
Δt: duration of a cycle.

3.1 The window dynamics

By integrating the expression of the equation (6), we get:

$$W(t) = \begin{cases} k_1 \exp(\frac{at}{T}) & \text{if } W < \hat{W} \\ a(\mu - \lambda/(1+a))t + k_2 & \text{if } W > \hat{W} \end{cases}$$

The constants are determined by considering the conditions at the limits: $W(0) = bW_{max}$ and $\hat{W} = k_1 \exp(\frac{a\hat{T}}{T}) = (\mu - \lambda/(1+a))\hat{T} + k_2$. We get $\hat{T} = \frac{T}{a}\ln(\frac{\hat{W}}{bW_{max}})$ so the window evolution is:

$$W(t) = \begin{cases} bW_{max} \exp(\frac{at}{T}) & \text{if } W < \hat{W} \\ a(\mu - \lambda/(1+a))(t - \hat{T}) + \hat{W} & \text{if } W > \hat{W} \end{cases} \tag{9}$$

$\hat{W}$ is given by $\hat{W} = (\mu - \lambda)T$. Hence, we obtain an expression of the window depending on the parameters μ, λ, T and B.

3.2 the cycle duration expression

We have to find the expression for the two cases:

- $\hat{W} > bW_{max}$ In this case we have the exponential and the linear phase $W_{max} = a(\mu - \lambda/(1+a))(\Delta t - \hat{T}) + \hat{W}$. Then,

$$\Delta t = \frac{W_{max} - \hat{W}}{a(\mu - \lambda/(1+a))} + \frac{T}{a}.\ln(\frac{\hat{W}}{bW_{max}}) \tag{10}$$

- $\hat{W} < bW_{max}$ The window here increment just linearly, then k_2=bW_{max}, and

$$\Delta t = \frac{W_{max}(1-b)}{a(\mu - \lambda/(1+a))} \tag{11}$$

By comparing our expressions derived for Δt with the one of T. Kelly in his paper [5], we conclude that they are quite close for all μ, λ, and T for small B. But with a large buffer, the difference is about some seconds for low delays. Furthermore, the constant of T. Kelly is $(-\ln(b)/\ln(1+a))$, which gives a $\Delta t = (-\ln(b)/\ln(1+a))T$. This last expression is obtained by calculating the duration that it takes W in the equation $W = bW_{max}\exp\frac{t}{100T}$ to reach W_{max} starting from the initial value of bW_{max}.

3.3 The average throughput expression

The average throughput equation is given by:

$$\overline{thp} = \frac{N_1 + N_2}{T_1 + T_2} \tag{12}$$

the number of packets N of a cycle is: $N_i = \int_{t_{i-1}}^{t_i} thp_{out}(t)dt$. Here also, we have to consider the two cases:

- $\hat{W} > bW_{max}$. For the first phase, with a duration of $\hat{T}$, we have the following equation: $thp_{out} = W(t)/T$. Then,

$$N_1 = \int_0^{\hat{T}} \frac{W(t)}{T}dt = \int_0^{\hat{T}} \frac{bW_{max}exp(\frac{at}{T})}{T}dt = \frac{b}{a}W_{max}(exp(\frac{a\hat{T}}{T})) = \frac{\hat{W} - bW_{max}}{a} \tag{13}$$

For the second phase, with a duration of $\Delta t - \hat{T}$, we have the following equation: $thp_{out} = \mu - \lambda/(1+a)$. Then,

$$N_2 = \int_{\hat{T}}^{\Delta t} (\mu - \lambda/(1+a))dt = (\Delta t - \hat{T})(\mu - \lambda/(1+a)).$$

The average throughput of the first case is:

$$\overline{thp} = \frac{\frac{\hat{W} - bW_{max}}{a} + (\Delta t - \hat{T})(\mu - \lambda/(1+a))}{\Delta t} \tag{14}$$

- $\hat{W} < bW_{max}$. In this case there is only one linear phase, and we have the following average throughput: $\overline{thp} = \mu - \lambda/(1+a)$.

By comparing the two expressions of the throughput, we find that the second case with a linear growth gives us the better throughput; this is an obvious result since with the linear growth, the queue is never empty. But the interesting point reside in knowing which parameters of the system that gives us the better throughput. We discuss this point in the next section.

Figures 2-4 present the variation of the throughput with the parameters λ, T and B with a fixed $\mu = 50$Mbps. Fig 2 with a value of $B = 10$ corresponds to the first case where the window has an exponential growth.

The STCP throughput in Fig 2-3 is normalized. It represents the proportion of the use of the available bandwidth $\mu - \lambda$ given for the STCP. By observing, the graphs, we see that the STCP performs better for high λ (in the sense that it achieves a larger portion of $(\mu - \lambda)$. Furthermore, for large λ the throughput proportion can go beyond one. This is due to the exponential factor of growth a that enters in the expression of the throughput and let it go beyond 1 by a factor of $1 - (\mu - \lambda/(1+a))/(\mu - \lambda)$.

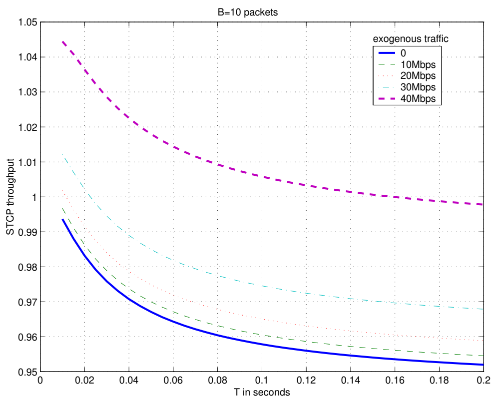

Figure 2: The throughput of STCP in presence of exogenous traffic with B=10 for various values of λ

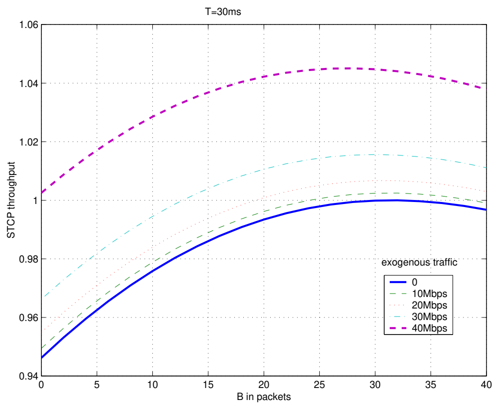

Figure 3: The throughput vs B for $T = 30$ for various values of λ

Fig 4 plots eq. (14) which gives also the global throughput up to the some limit value of $B = 86$ beyond which (14) does not provide the total throughput (the global throughput for larger buffer sizes is then constant given by $\mu - \lambda/(1+a)$.

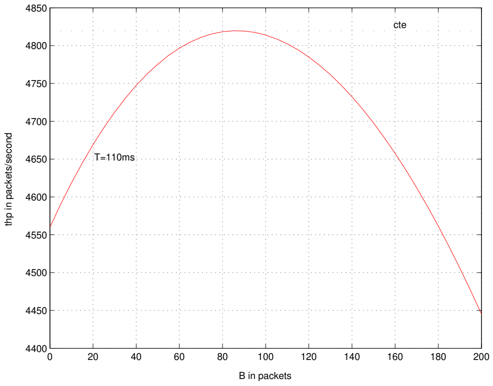

Figure 4: The throughput vs Buffer size for $T =$ 110ms for $\lambda = 10Mbps$

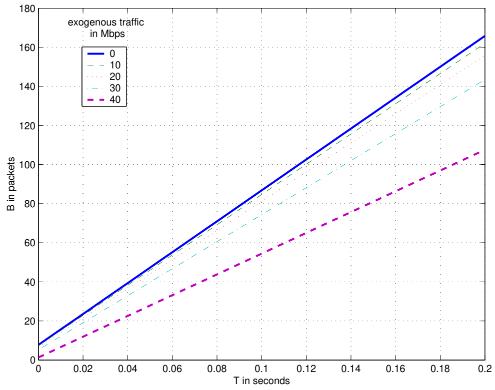

Figure 5: The critical Buffer size when $\mu =$ 50Mbps for λ between 0 and 40 Mbps

4 The condition for an AIMD type behavior

We shall discuss the equations derived from the critical equality: $\hat{W} = bW_{max}$. It yields

$$B_{th} = \frac{\mu(T(\mu-\lambda) - (1+a)b((\mu - \lambda/(1+a))(T + a\tau_1) + \lambda/((1+a)\mu)))}{b(1+a)(\mu - \lambda/(1+a))} \tag{15}$$

For $B > B_{th}$ we have only the linear phase in which STCP behaves like an AIMD protocol. We also obtain then the best throughput. We show this condition in the figure 5.

Also, by applying the main condition to the parameter b of STCP, we see that we can obtain an optimal b depending on μ, λ, T and B. We took two cases for the condition with b: the first is for $B = BDP$, and we remark that b doesn't converge to 1 when B approach to 0. Conversely, when we took $B < BDP$, we obtained the convergence. Both cases are for $\mu = 10$Mbps.

5 Comparison between the STCP and TCP Reno

Firstly, we derive the expression of the window for the TCP Reno, then we proceed like the section (3); we can refer also, to the paper [3]. We summarize here the results obtained.

5.1 Equations of the TCP Reno

We assume that there is no slow start phase. Hence, the equations are for the congestion avoidance phase.

$$\frac{dW(t)}{dack} = \frac{1}{W} \ \forall\ W(t) \tag{16}$$

$$thp_{out} = \begin{cases} \frac{W(t)}{T} & \text{if } W < \hat{W} \\ \mu - \frac{\lambda}{1+(1/W)} & \text{if } W > \hat{W} \end{cases} \tag{17}$$

Assuming that $W >> 1$ (when considering wide area networks) the last case in thp_{out} can be approximated by $\mu - \lambda$. Let $\hat{W} := T(\mu - \lambda)$.

- case 1: $\hat{W} > \frac{W_{max}}{2}$. We have [3]

$$W(t) = \begin{cases} \frac{1}{T}t + \frac{W_{max}}{2} & \text{if } W < \hat{W} \\ \sqrt{2}[(\mu-\lambda)(t-\hat{T}) + \frac{\hat{W}^2}{2}]^{\frac{1}{2}} & \text{if } W > \hat{W} \end{cases} \tag{18}$$

- case 2: $\hat{W} < \frac{W_{max}}{2}$. We have $W(t) = (2(\mu-\lambda)t)^{1/2}$. With $\hat{T} := T(\hat{W} - \frac{W_{max}}{2})$ we get

$$T_{max} = \begin{cases} \frac{W_{max}^2 - \hat{W}^2}{2(\mu-\lambda)} + \hat{T} & \text{if } \hat{W} > \frac{W_{max}}{2} \\ \frac{W_{max}^2}{2(\mu-\lambda)} & \text{if } \hat{W} < \frac{W_{max}}{2} \end{cases} \tag{19}$$

Let δt denote the cycle duration, i.e. $T_{max} = \delta t$. The average throughput is

$$\overline{thp} = \begin{cases} \frac{1}{T_{max}}\left(\frac{\overline{T}^2}{2T^2} + \frac{\overline{T}W_{max}}{2T} + (\mu-\lambda)(T_{max}-\hat{T})\right) & \text{if } \hat{W} > \frac{W_{max}}{2} \\ \mu - \frac{\lambda}{1+(1/W)} & \text{if } \hat{W} < \frac{W_{max}}{2} \end{cases} \tag{20}$$

An simpler approximate expression is given by:

$$\overline{thp} = \begin{cases} \frac{3W_{max}^2}{8T_{max}} & \text{if } \hat{W} > \frac{W_{max}}{2} \\ \mu - \frac{\lambda}{1+(1/W)} & \text{if } \hat{W} < \frac{W_{max}}{2} \end{cases}$$

5.2 Comparison of the throughputs

TCP Reno has also $B_{th} = \mu T - (\lambda + 2\mu)/(\mu - \lambda)$ and for $B < B_{th}$ we have two phases: the linear growth (corresponding to an empty queue) and a sub-linear growth (corresponding to a nonempty queue). By combining the two conditions of STCP and TCP Reno, we conclude that with small buffer size we are always within the case of having two phases (i.e. within the possibility of having an empty queue).

We present some examples. We take $\mu = 50$Mbps. In the figure 6, we show the variation of the ratio $(\overline{thp}_{STCP})/(\overline{thp}_{TCP})$ versus T for some values of B.

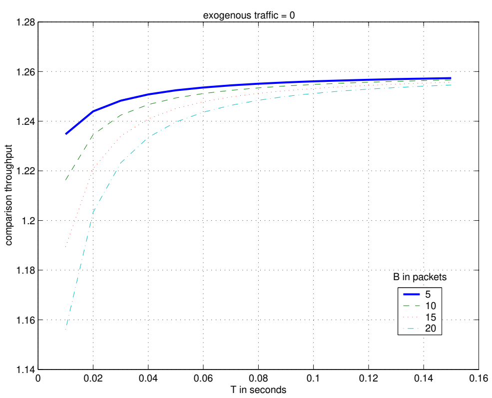

Figure 6: Comparison between the average throughput of STCP and TCP Reno for λ between 5 Mbps and 20 Mbps, for $B < B_{th}$

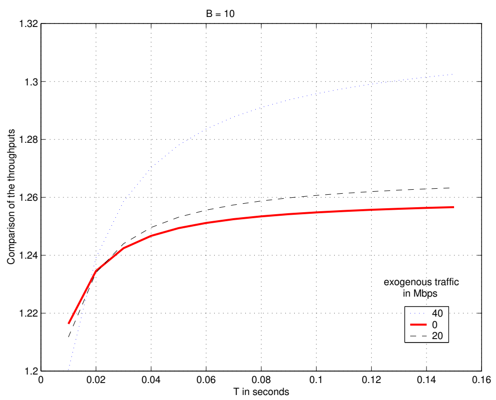

Figure 7: Comparison between the average throughput of STCP and TCP Reno for λ=0,20,40 Mbps

We see from the graph that STCP gives a better throughput and with the lowest buffer size we have the largest difference between it and TCP Reno. But with the case of the one phase the throughputs are very close, or equal (refer to the equations), it depends on the maximum window size.

As the figure 7 shows, the STCP performs much better with large λ and large delays comparing with the TCP Reno. On the other hand, when the delays are small it is the opposite.

5.3 Comparison between the period times Δt and δt

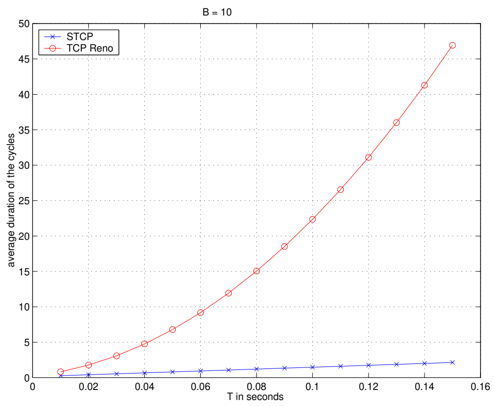

Figure 8: Comparison between the cycled durations Δt and δt for $\lambda = 0$

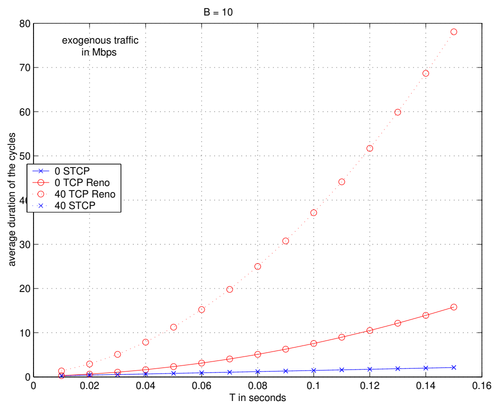

Figure 9: Comparison between the cycle durations Δt and δt for λ=0 and 40 Mbps.

Figures 8 9, 10 illustrate the difference between the cycle times Δt and δt. We observe that the frequency of cycles (and thus of losses) in STCP is very large compared to the TCP Reno.

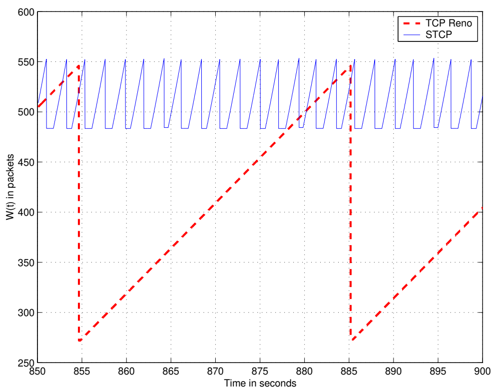

Figure 10: Comparison of STCP and TCP Reno: Window variation vs the time for μ = 50Mbps, λ = 10Mbps, T = 110ms and B = 20packets. We obtain the same W_{max} corresponding to a full pipe.

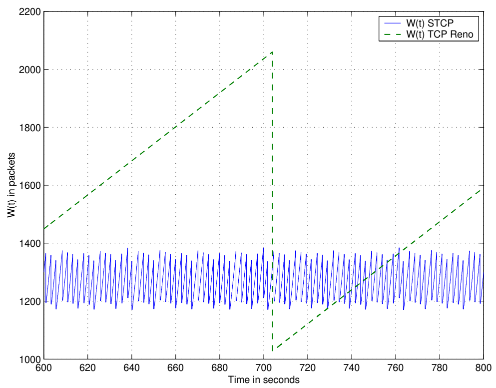

Figure 11: Comparison of STCP and TCP Reno: Window variation vs the time under condition (21) with $B = 15$ and $B_0 = 20.4$ for $\lambda = 0, \mu_1 = 300Mbps, \mu = 100Mbps, \tau_1 = 5msec, \tau_2 = 80msec$.

5.4 Link utilization

We have seen the link utilization of the two protocols in previous paragraphs, but here we illustrate it with the B/BDP parameter;
for $\lambda = 0$, $\mu = 50$Mbps and $T = 40$ms we present our findings in Table 1 below.

B/(μT)	STCP	TCP Reno
0.04	0.97	0.78
0.08	0.99	0.81
0.16	1	0.85
0.32	1	0.89

Table 1: Link utilization with different B/BDP values

From these values (which we shall compare with simulation), we conclude that STCP uses the link more efficiently than TCP Reno. Figure 10 shows traces of TCP Reno and STCP with the same parameters. It is seen indeed that a better utilisation is obtained under STCP and that the cycle time for STCP is shorter.

6 Simulation Results

We show here in the table 2 some simulation results for the throughput and some analytic values. We define here μ_1 the bandwidth of the link between the source and the buffer. The $\overline{thp}$ is gave in packets/s with a packet size equal to 1040 Bytes.

	Parameters						$\overline{thp}$ STCP			$\overline{thp}$ TCP Reno			ratio	
Case	B	μ_1	μ	λ	τ_1	τ_2	Ana.	Simu.	%err.	Ana.	Simu.	%err.	Ana.	Simu.
1	20	100	50	10	5	50	4669	4544	2.6	3725	3646	2.1	1.25	1.24
2	15	100	50	0	5	50	5789	5636	2.6	4620	4101	11.2	1.25	1.37
3	10	100	50	0	5	50	5756	5597	2.7	4587	4498	1.9	1.25	1.24
4	10	300	50	0	5	70	5737	5266	8.9	4566	4101	10.1	1.25	1.28
5	5	100	50	0	5	10	5814	5711	1.7	4674	4508	3.5	1.24	1.26

Table 2: Some simulations results

Simulations 1-5 in the table 2 confirm our analytic results providing less than 10% errors.

7 Some extreme cases

We observed in some simulations, especially for large buffer sizes and BDP, that many time outs occurred. We do not include this in our analysis, which would still hold in MIMD congestion control that can recover from congestion without time-outs.

The other extreme is of very small buffers. In the case of Tahoe, small buffers resulted in losses in the slow-start phase [1, 3]. In our case the main change with respect to our analysis will be that W_{max} is smaller than (7). We'll estimate the correct value of W_{max} and compute the buffer size B_0 below which it holds assuming $\lambda = 0$.

Let W_n be the window size at the nth RTT (round trip time) where W_0 is the initial value just after the previous loss. Then $W_{n+1} = (1 + a)W_n$ till a loss occurs. In each RTT the queue builds up and then empties. While the queue is nonempty, ACKs are generated at a rate of μ so that packets in the next RTT are generated at a rate of $(1 + a)\mu$. This is larger than the link capacity μ so that the queue builds up at a rate of $a\mu$. The maximum queue size attained at the nth RTT is then $Q_n = aW_{n-1}$.

We say that an early loss occurs if

$$B < a\mu T. \tag{21}$$

This condition states that starting from an empty queue, the time $B/(a\mu)$ it takes to fill the queue (at the rate $a\mu$) is shorter than T. It will then take n^*T for the loss to occur where $n^* = \max_m\{Q_m < B\} = \max_m\{aW_{m-1} < B\} = \lfloor B/a \rfloor$. Thus the loss will be detected when

$$W_{max} = \lfloor B/a \rfloor + 1. \tag{22}$$

When (21) holds then the throughput of STCP may be much lower than of TCP Reno. An example is given in Fig. 11. It gives $W_{max} = 1360$ through simulations where as (7) predicts a value of 2080 (42% error). (22) in contrast gives a value of 1515 for which the error is reduced to around 10%.

We classify (21) as an extreme case since for scalable TCP it applies when B is smaller than 1% of the BDP.

8 Conclusions

We have analyzed in this paper the performance of STCP that shares a bottleneck link with an exogenous non-controlled traffic. We presented a fluid analysis which provided the throughput. We identified conditions under which the window evolution of STCP behaves as an AIMD protocol (in terms of time). We finally validated our model through simulations and identified some extreme cases that are not covered by our model.

References

[1] E. Altman, F. Boccara, J. Bolot, P. Nain, P. Brown, D. Collange,C. Fenzy "Analysis of the TCP/IP Flow Control Mechanism in High-Speed Wide-Area Networks", *IEEE CDC*, New Orleans, USA, 368-373, Dec 2005.

[2] S. Floyd. HighSpeed TCP for Large Congestion Windows. RFC 3649, Experimental, December 2003. Available at www.icir.org/floyd/hstcp.html.

[3] T. Lakshman U. Madhow, "Window-based congestion control for networks with bandwidth-delay products and random loss: a study of TCP/IP performance" , *Proc HPN'94 (High Performance Networking)*, pp.133-147, Grenoble, France, Jun. 1994.

[4] C. Jin, D. X. Wei and S. H. Low, "Fast TCP: motivation, architecture, algorithms, performance", *Proc. of IEEE Infocom*, March 2004.

[5] T. Kelly, "Scalable TCP Improving Performance in HighSpeed Wide Area Networks", First International Workshop on Protocols for Fast Long-Distance Networks, Geneva, Feb 2003.

[6] S. McCanne and S. Floyd. ns Network Simulator. Available at http://www.isi.edu/nsnam/ns/.

[7] L. Xu, K. Harfoush, and I. Rhee, "Binary Increase Congestion Control (BIC) for Fast Long-Distance Network," in *Proc. of IEEE Infocom*, March 2004.

[8] G. Vinnicombe, "On the stability of networks operating TCP-like congestion control", *Proceedings of the IFAC World Congress*, Barcelona, Spain, 2002.