Analysis of TCP with Several Bottleneck Nodes

Chadi Barakat, Eitan Altman

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Analysis of TCP with Several Bottleneck Nodes

Chadi Barakat — Eitan Altman

N° 3620

February 1999

THÈME 1

ISSN 0249-6399

INRIA
SOPHIA ANTIPOLIS

Analysis of TCP with Several Bottleneck Nodes

Chadi Barakat , Eitan Altman

Thème 1 — Réseaux et systèmes
Projet Mistral

Rapport de recherche n° 3620 — February 1999 — 25 pages

Abstract: Many works have studied the performance of TCP by modeling the network as a single bottleneck node, the one having the slowest outgoing rate on the path. In these works, the behavior of the protocol, especially during slow-start, has been characterized as a function of this main bottleneck. In this paper, we present a more general model taking into account all the nodes on the path. We show that, in addition to the main bottleneck, the performance of the protocol can be seriously affected by the parameters of the other nodes. The other nodes may, in some cases, cause an improvement in the performance since they may decrease the burstiness of traffic arriving at the main bottleneck. They could, on the other hand, cause performance degradation in case losses occur in those nodes. Indeed, if the buffers in these nodes are not well dimensioned, then the congestion can be shifted to them even if their outgoing rate is faster than that of the main bottleneck. The effect of the other nodes on the behavior of TCP during Congestion Avoidance can be neglected. However, we show analytically that the slow-start behavior predicted by a single node model can be completely different when considering the general model. Our analysis is followed by guidelines for the dimensioning of network buffers so as to improve the performance of TCP. Also, as a result of this general model, we are able to identify, understand and analyze a problem in the operation of TCP which results in a throughput deterioration and which cannot be explained by a single node model. By simulations, we show that this problem can be resolved if RED (Random Early Detection) buffer management policies are deployed in the Internet.

Key-words: TCP, Slow-Start, Congestion Avoidance, Modeling, Simulations, Buffer Management, RED, Drop Tail.

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles, B.P. 93, 06902 Sophia Antipolis Cedex (France)
Téléphone : 04 92 38 77 77 - International : +33 4 92 38 77 77 — Fax : 04 92 38 77 65 - International : +33 4 92 38 77 65

Analyse de TCP avec plusieurs noeuds

Résumé : Plusieurs travaux ont étudié la performance du protocole TCP en modélisant le réseau avec un seul noeud, celui ayant la plus petite bande passante résiduelle le long du chemin entre la source et la destination. Dans ces travaux, le comportement du protocole, particulièrement durant la phase Slow-Start, a été caractérisé en fonction des paramètres de ce noeud appelé goulot d'étranglement principal. Dans notre papier, on présente un modèle plus général tenant compte de tous les noeuds. Nous montrons qu'en plus du goulot principal, la performance de TCP est une fonction des paramètres des autres noeuds. Ces noeuds peuvent améliorer la performance en réduisant la sporadicité du trafic TCP à l'entrée du goulot principal. Mais en même temps, ils peuvent causer une dégradation de performance lorsque des pertes apparaissent dans ces noeuds. En effet, si les tampons dans ces noeuds ne sont pas bien dimensionnés, la congestion peut apparaître dans un d'eux même s'ils ont plus de bande passante résiduelle que le goulot principal. L'effet de ces noeuds sur le comportement de TCP durant la phase Congestion Avoidance peut être négligé. Cependant, notre analyse montre que le comportement durant Slow-Start peut être complètement différent de celui prévu par un modèle considérant seulement le goulot principal. Cette analyse aide à trouver le meilleur dimensionnement des tampons afin d'améliorer la performance de TCP. Enfin, ce modèle nous permet d'identifier et d'analyser un problème dans le fonctionnement de TCP qui cause une détérioration du débit. Avec des simulations, nous montrons que ce problème peut être résolu si la stratégie de gestion des tampons RED (Random Early DEtection) est déployée dans les noeuds du réseau.

Mots-clés : TCP, Slow-Start, Congestion Avoidance, Modélisation, Simulations, Gestion des Tampons, RED, Drop Tail.

1 Introduction

TCP (Transmission Control Protocol) is used by many applications of today's Internet to control the flow of their packets according to network congestion. Because of its crucial role in the operation of the Internet, the performance of this protocol has been intensively studied [2, 5, 6, 10, 11]. The analytical models developed assume that the network crossed by the connection can be modeled as a single bottleneck node, the one having the lowest outgoing rate on the path between the source and the destination. This model is correct if the buffer capacity and the available bandwidth in the other nodes crossed by the TCP connection are very large compared to those of the node chosen as a a single and main bottleneck. By available bandwidth in a node we mean the rate at which TCP packets can leave this node towards the destination. However, due to the fluctuations in real networks, these quantities can be very close which may affect the behavior of TCP and may result in a different performance than that predicted by the single bottleneck node model. As we will show later, this effect can be ignored during congestion avoidance but it can change completely the behavior of the slow-start algorithm. The bursts sent during slow-start can cause a queue building in many nodes not only in that taken as the main bottleneck. This distribution of the queue on multiple nodes instead of on a single one can avoid a buffer overflow predicted by the single bottleneck model which results in an improvement in the performance. But also it can cause an unpredicted overflow if the buffers in the other nodes are not suitably dimensioned. This may happen even if the buffer in the node taken as the main bottleneck is chosen accordingly to the condition found in [2, 11] so as to avoid losses in the slow-start phase. It is well known that a packet loss due to buffer overflow during slow-start results in a smaller window at the beginning of the congestion avoidance phase and then in a lower throughput [3, 4].

In this paper, we study the performance of TCP as a function of the parameters of all the nodes crossed by the connection. As parameters, we consider the available bandwidth and the buffer capacity. Except when mentioned, the capacity of a buffer is managed according to the most common Drop Tail policy. We present a general model consisting of many bottlenecks and then we show that it can be simplified to a one with two nodes without changing the performance of the protocol. The operation of TCP is characterized with such model and the analytical results are validated by a set of simulations using NS, the Network Simulator [13]. Among our results, we show that to avoid losses during the slow-start phase and therefore to improve the performance of the protocol, the buffering capacity in the other nodes must scale linearly with that in the main bottleneck. Although the buffer requirement in these nodes is not so stringent as in the main bottleneck, they are necessary to absorb the bursts sent during slow-start when the output rate of their nodes falls below twice the available bandwidth in the main bottleneck.

Finally, an important problem in the operation of TCP is identified. This problem, which is one of the results of the consideration of the other nodes on the path, corresponds to the loss of the packet retransmitted directly after the detection of a loss. This second loss of the

packet forces TCP to wait a long timeout to detect it and makes the congestion avoidance phase start at a very small window which results in a poorer performance. The appearance of such phenomena is analytically studied and simulations are conducted to prove that it can be resolved if other buffer management policies are adopted in the network.

In the next section, we present our general model. In section 3 and 4, we study the behavior of TCP as a function of the parameters of all the nodes on the path. In section 5, we prove that this behavior can be predicted using a simplified model of two bottleneck nodes. This simplified model is used in section 6 to evaluate the performance of the protocol. In section 7, we analyze the problem of the retransmission loss and in section 8 we conclude the paper.

2 The multiple node network model

In this section, we argue the need to consider, in addition to the node with the lowest outgoing rate, the parameters of some of the other nodes crossed by the connection in order to understand the behavior of TCP. First, we present a brief overview of TCP and its two algorithms Slow-Start and Congestion Avoidance, then we present our model. The performance of the protocol with this model is studied in the next two sections.

2.1 Overview of TCP

TCP provides applications with an end-to-end byte streaming reliable service. Reliability and congestion control are based on the information carried by the acknowledgments (ACK) sent by the receiver and on retransmission timers set upon packet transmission. A cumulative ACK, sent by the destination in response to a data packet, tells the source the identity of the last in-order packet received. The source maintains a window variable, denoted W in the sequel, which represents the maximum number of packets that it can send without the reception of any ACK. An incoming ACK with a new information triggers the transmission of new data and makes the source increase its window. Now, if the retransmission timer expires before the reception of a new ACK or if the source receives three ACKs carrying the same information (called duplicate ACKs), it concludes that the packet is lost as a result of a congestion in the network. Thus, it reduces its window and it enters a recovery phase. The detection of a loss by means of duplicate ACKs is called the Fast Retransmit algorithm [14]. Slow-start and Congestion Avoidance are the two algorithms designed by Jacobson [9] to change the window size as a function of network conditions. Slow-start is used to increase W gradually from 1 to reach the source estimation of the network capacity. At this point, called the slow-start threshold W_{th}, the source switch to congestion avoidance. During slow-start, W is increased by one for every new ACK which results in an exponential growth of the window and the transmission of bursts of packets. This gradual increase,

which results in a lower burstiness when compared to a direct transmission at a window equals to W_{th}, prohibits the source from overwhelming the network with a large burst of packets. During congestion avoidance, the window is increased slowly by at most one packet every Round Trip Time (RTT). This slow growth aims to probe the network for any extra bandwidth. When a loss occurs at a window W, the source considers that the network is congested and it estimates the available capacity as half this window, thus it sets W_{th} to $W/2$. At the beginning of the connection however, the source doesn't have any information on the network, so it sets W_{th} to a default value usually equals to the window advertised by the receiver.

2.2 The general model

Let μ be the smallest available bandwidth on the path between the source and the destination. If we suppose that the return path is not congested and that the receiver acknowledges instantaneously every packet, μ can then be considered as the rate at which ACKs return to the source. Two consecutive ACKs are separated by at least $1/\mu$. Because all the nodes between the bottleneck μ and the destination receive packets at a rate slower than the available bandwidth on their output links, their parameters don't affect the performance of TCP. However, according to TCP congestion control algorithms [9, 14], the source can sometimes send a burst of two packets in response to an incoming ACK. This happens during slow-start and when the window is incremented by one during congestion avoidance. If the nodes between the source and the bottleneck μ have an available bandwidth larger than 2μ, these bursts will cross these intermediate nodes as if they don't exist. In this case we can suppose that the bottleneck μ is fed directly by the source and consequently we get the same performance as that predicted by the single bottleneck node model. Now, if one of these intermediate nodes has an outgoing rate slower than 2μ, the bursts will be slowed which results in a queue building in this node. A TCP packet following a burst of two packets by a time $1/\mu$ will find this node serving the second packet of the burst. We say here that the bursts of the TCP source are partially absorbed by this node which reduces the queue building rate at the bottleneck μ. This partial absorption can overflow the buffer of this intermediate node and it can also avoid a buffer overflow in another node. Thus, any network modeling must take account of the nodes preceding μ and having an outgoing rate slower than 2μ.

In figure 1 we show our model where, in addition to the bottleneck μ, n nodes of buffers B_i and of outgoing rate μ_i are considered. To affect the performance of the protocol, the μ_i must satisfy

$$\mu < \mu_n < \mu_{n-1} < \cdots < \mu_1 < 2\mu$$

Thus links become slower downstream. For analysis purposes this assumption is without loss of generality, since if a down link is faster than its predecessor link then no queueing will occur there, so the related node can be ignored when analyzing the performance of the

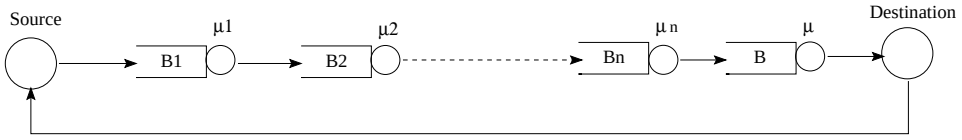

Figure 1: The multiple node network model

connection. The restriction $\mu_1 < 2\mu$ is without loss of generality as well, since a link rate μ_1 larger than 2μ would mean that a queue never builds up in the first node, since the input rate to that queue, even at bursty periods, is upper-bounded by 2μ.

3 The behavior of TCP during Congestion Avoidance

Let T be the minimum Round Trip Time which is the sum of the two-way propagation delay and the service time in the different nodes. We suppose that, during congestion avoidance, the window increases by exactly one packet every RTT [1]. After the transmission of a burst of two packets as a result of the window increase from $W-1$ to W, the source transmits $W-1$ packets at a rate μ followed by the transmission of a new burst of two packets when the Wth ACK is received. Here, the window is again increased to $W+1$. If W is smaller than the pipe size μT, the time between the transmission of the two bursts is equal to T which is long enough for all the nodes to serve the packets sent between them. Thus, no queue builds up and the window grows linearly by one packet every T.

When the window exceeds the pipe size, bursts of packets are followed after $1/\mu$ by another packet. Thus, there will not be enough time for the nodes of rate slower than 2μ to serve the two packet of the burst. Then, a queue starts to build up in the network. The packet in excess of the network capacity must wait somewhere on the path. According to the single bottleneck model, this queue is seen only in node μ and the congestion avoidance phase ends when buffer B overflows. This happens at a window $W = W_{max} = B + \mu T$. Such assumption is true if the preceding nodes (1 to n) are able to serve the packets sent between two bursts so that a new burst finds the buffers in these nodes empty. By empty we mean that no TCP packets of the same connection are waiting in these buffers. If this burst finds a non empty buffer other than in node μ, a queue builds up in this buffer in addition to that in node μ which results in a different value of W_{max}, thus in an another throughput of the connection than the one obtained in the simplified single node model.

Consider the two bursts sent when the window increases from $W-1$ to W and from W to $W+1$. Because we are interested in the case $W > \mu T$, the time between the transmission of these two bursts is equal to W/μ and the number of packets to be served during this time is

[1] This requires an increase in W by $1/\lfloor W \rfloor$ upon the receipt of a new ACK.

equal to $W+1$. We can say that at a window W, the source transmits packets at an average rate $R_{CA} = \frac{W+1}{W}\mu$. It is clear that this rate is always greater than μ when W increases and then the number of waiting packets in the network always increases during congestion avoidance. These waiting packets are distributed between node μ and the nodes from 1 to n having an available bandwidth smaller than R_{CA}. This distribution makes the queue in node μ build up at a rate slower than one packet per RTT as long as there is a μ_i satisfying $\mu_i < R_{CA}$. If such μ_i exists, the queue building rate at μ is constant and equals to $\mu_1 - \mu$. Given that R_{CA} is a decreasing function of W, the effect of nodes μ_i on the performance decreases when W moves away from μT. Once we reach a window that results in a R_{CA} greater than all the μ_i, all the waiting packets move to buffer B which leads us to the case of the single bottleneck node model. We see well that, due to this partial absorption of the bursts, W can reach values greater than W_{max} which results in a better performance although the buffer in node μ doesn't change. But this may also deteriorate the performance by making the congestion avoidance phase end at a window smaller than W_{max}. This happens when the overflow occurs in a node μ_i with a small B_i.

Because R_{CA} is very close to μ, the only case of interest is when there exists a μ_i equal to μ. Let μ_i be the closest node to the source having an available bandwidth equal to μ and suppose that all the nodes between the source and node μ_i have an available bandwidth greater than R_{CA}. Therefore, only node μ_i in the network is fed at a rate (R_{CA}) greater than its service rate. The bursts are then completely absorbed by this node and a queue never builds up in the other nodes including those between it and the destination. Thus, if many links in the network have the same available bandwidth μ, the queue builds up during congestion avoidance in the input buffer of the closest link to the source. The capacity of this buffer determines alone the value of W_{max}. In the following, we suppose that the μ_i are large so that the queue builds up only in node μ (i.e. there is no $\mu_i < R_{CA}$). Also, B in figure 1 represents the buffer capacity of the closest node to the source having an available bandwidth equal to μ.

To illustrate our conclusions, we simulate a scenario where a TCP connection crosses two bottleneck nodes of rate μ_1 and μ. T is taken equal to 560ms (the connection is routed through a GEO satellite) and μ to 1.5 Mbits/s (T1 link). TCP packets are of total size 512 bytes. The buffers in the two nodes are set to $B_1 = 100$ and $B = 50$ packets. First we take μ_1 equal to μ, then we increase it slightly in order to get $R_{CA} < \mu_1$. We plot in figure 2 the window variation as a function of time for $\mu_1 = 1.5$ Mbits/s and $\mu_1 = 1.6$ Mbits/s. We see well that, although we increase the rate μ_1, the average window size decreases resulting in a decrease in the throughput from 1.371 Mbits/s to 1.247 Mbits/s. Indeed, the increase in μ_1 has moved the queue from B_1 to B. Given that B is smaller than B_1, W_{max} decreases from $(B1 + \mu T)$ to $(B + \mu T)$ which explains the throughput deterioration.

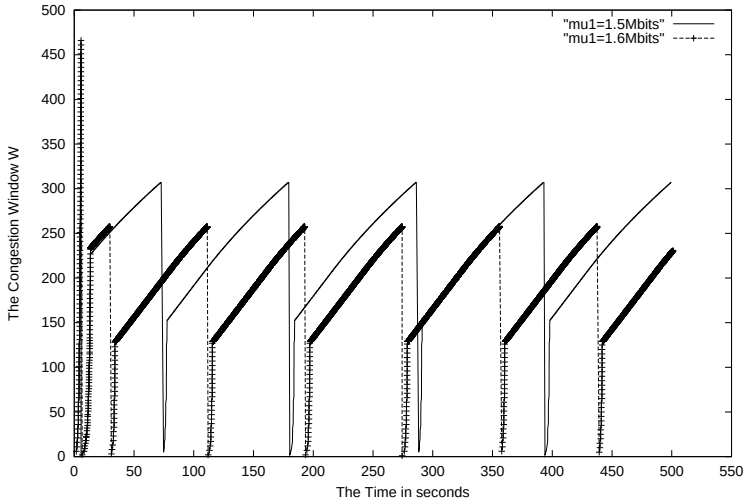

Figure 2: The performance as a function of μ_1

4 The behavior of TCP during Slow-Start

During slow-start, an new incoming ACK causes an increase in the window by one and triggers the transmission of a burst of two packets. This algorithm is called at the beginning of the connection and when a loss is detected by a timeout. In addition to the gradual increase in the window size, this algorithm serves to activate the ACK clock required for the operation of TCP and lost when a timeout occurs. However, the Tahoe version of TCP resorts always to slow-start even if a loss is detected with three Duplicate ACKs. Resorting always to slow-start is a conservative behavior because it supposes that a loss is always an indication of a severe congestion in the network. To study the behavior of this algorithm with our model, we consider the Tahoe version of TCP where slow-start appears frequently. The analysis can also be applied to the other versions of TCP whenever a slow-start is called. The only difference is that the other versions try to recover from the losses during slow-start using a normal recovery phase. But, a buffer overflow during slow-start causes the loss of many packets from the same window of data, thus these versions fail most of time [6] and resort finally to slow-start after a long timeout.

In contrast to congestion avoidance, the effect of nodes 1 to n on the performance is more pronounced during slow-start. The source sends long bursts of packets at twice the available bandwidth μ due to a burst of two packets in response to every new ACK. Thus, it is more likely to find nodes between the source and the bottleneck μ that have an available bandwidth between μ and 2μ than between μ and R_{CA} as in the previous section. If such nodes

exist, a burst sent at rate 2μ causes also a queue building in these nodes and, depending on the capacity of their buffers, an overflow may happen in any node not only in the bottleneck μ. This overflow results in a double consecutive slow-start phases and is the result of a small buffering capacity compared to the bandwidth delay-product of the network (in [3], we have proved that, for even smaller buffers, a cyclic phenomena of three consecutive slow-start phases may appear yielding a further deterioration of the performance). The double slow-start problem occurs when the threshold W_{th}, which is equal to $W_{max}/2 = (\mu T + B)/2$ in our case, is very large so that the network buffers fail to absorb the bursty traffic sent during slow-start before getting in congestion avoidance. A condition for this phenomena has been calculated in [2, 11]. This condition accounts only for the bottleneck node μ. In this section, we will try to recalculate this condition in the presence of multiple nodes between the source and the destination as in figure 1.

The slow-start phase is divided into mini-cycles of duration T [11]. During mini-cycle n, the source sends a burst of 2^n packets at rate 2μ. Mini-cycle $n+1$ starts when the ACK for the first packet of this burst reaches the source. These bursts propagate from node to node and create queues in the different buffers of figure 1. The queue building rates are given by

$$\mu_{i-1} - \mu_i \quad \text{in buffer } B_i \ (i = 1 \dots n) \text{ with } \mu_0 = 2\mu$$

$$\mu_n - \mu \qquad \text{in buffer } B$$

Given these rates, we can calculate the number of packets sent in a burst and required to overflow each buffer

$$\left.\begin{array}{ll} S_i = \frac{\mu_{i-1}}{\mu_{i-1}-\mu_i} \times B_i & \text{for buffer } B_i \ (i = 1 \dots n) \text{ with } \mu_0 = 2\mu \\ S = \frac{\mu_n}{\mu_n - \mu} \times B & \text{for buffer } B \end{array}\right\} \quad (1)$$

We get a loss in the slow-start phase due to buffer overflow, therefore a second slow-start and a small window at the beginning of the congestion avoidance phase, if in a given mini-cycle the source sends a burst of packets at rate 2μ larger than at least one of the S_i and S. If this is the case, the overflow is detected by the source during the next mini-cycle, a new slow-start is called (recall that we are working with a Tahoe version which implements the Fast Retransmit algorithm) and the first packet retransmitted is the one lost at the buffer requiring the minimum number of packets to be overflowed. Let

$$S_B = \min_{i=1\dots n} (S, S_i)$$

then the loss occurs in mini-cycle n_B given by

$$2^{n_B - 1} < S_B \leq 2^{n_B}$$

and the window size W_B at which this loss happens is

$$W_B = 2^{n_B - 1} + S_B/2 \Longrightarrow 2W_B = 2^{n_B} + S_B$$

Hence, the double slow-start condition can simply be written as $W_{th} > W_B$. In contrast to that found in [11, 2], we see well that this condition involves the parameters of all the nodes not only those of the bottleneck with the smallest service rate. In case of cross traffic in node i, μ_i represents the bandwidth available to TCP connection and B_i the share of TCP packets in the global buffer in node i. We notice that even if we supply the node at the input of the slowest link with a large buffer so as to avoid the double slow-start according to the condition found in [11] ($\beta = B/\mu T < 1/3$)[2], the other network parameters can result in a low W_B, thus in an occurrence of losses and a deterioration of the performance. Also, because its input rate is bounded by μ_n which is smaller than 2μ, the queue in buffer B builds up slowly resulting in a higher value of S. Depending on the values of S_i, this distribution of the queue on many nodes may avoid an overflow predicted by the condition $\beta < 1/3$ which results in a better performance.

Now, if the buffers are large enough so that $W_{th} < W_B$, the bursts sent during slow-start will not cause an overflow and the source will get in congestion avoidance with one slow-start per TCP cycle. The window then increases slowly from $W_{max}/2$ to W_{max} where a normal loss occurs. A new slow-start is called with $W_{max}/2$ as a threshold. Thus, we get the same behavior as with the single bottleneck node model, of course if $\beta > 1/3$.

5 A simplified model with two bottleneck nodes

From the previous analysis, we conclude that any model must consider node μ because it determines the connection behavior during congestion avoidance, particularly the value of W_{max}. However, to well predict the behavior during slow-start, the preceding nodes which have a service rate between μ and 2μ must also be considered. To simplify the analysis without changing the results, these nodes can be replaced by an *equivalent* one which results in a simple model consisting of two bottleneck nodes (figure 3). In this section, we argue the adoption of this model and we use it to study the effect of network parameters on TCP performance.

Nodes 1 to n are replaced by that having the smallest S_i, since the node with the smallest S_i is the candidate to have the first loss during slow-start. The buffer size B' and service rate μ' for the equivalent node are chosen as follows as a function of B_i and μ_i.

The output rate μ' of the equivalent node is taken equal to μ_n and we suppose that it is directly fed by the source. The reason is that we do not wish to change the behavior of the last node with the slowest link rate (the one with buffer size B and link rate μ), whose input rate in the original problem is bounded by μ_n. We choose an equivalent buffer size B' so that the new equivalent node which has an input rate bounded by 2μ and an output

[2] According to [11], a double slow-start occurs if B is smaller than one third the Bandwidth-Delay product of the network μT.

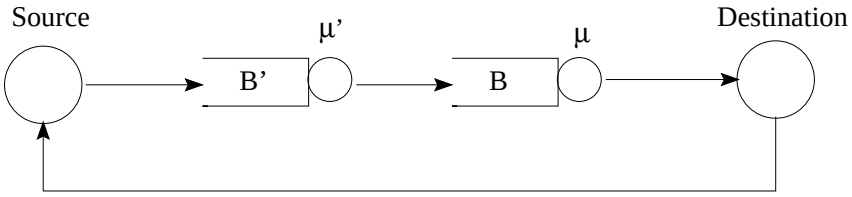

Figure 3: The simplified model

rate μ_n requires a burst of size $\min_{i=1\ldots n}(S_i)$ to be filled. Thus, $\mu' = \mu_n$ and B' is given by

$$S' = \frac{2\mu}{2\mu - \mu'} \times B' = \min_{i=1\ldots n}(S_i)$$

Before we pass to study the performance of TCP using this simplified model, we present some simulation results that prove the correctness of the equivalent node approach. We take for μ, T and packet size the same values as in the previous section and we give B the size of 20 packets. The different nodes are placed on the path between the source and the destination in a way so that the sum of the propagation delays of the links joining them is always equal to $T = 560$ ms. We vary n, the number of nodes preceding the bottleneck μ, between 1 and 10. For each n, we distribute the μ_i uniformly on the segment $[\mu, 2\mu]$. Thus, an μ_i is equal to

$$\mu_i = 2\mu - \frac{i}{n+1}\mu$$

A simple calculation shows that

$$S_i = B_i \times [2(n+1) - (i-1)] \text{ and } S = B \times (n+2)$$

If we take the B_i equal to B, then S_B will be always equal to S and the overflow will always occur in node μ. To change the network behavior, we choose the B_i so that to move the overflow to one of the nodes that precedes the bottleneck μ for some values of n. By giving the B_i the size 17 packets, we make S_B equal to S_n for the following n

$$S_n = 17(n+3) \ , \ S = 20(n+2) \ , \ S_n < S \implies n \geq 4$$

We can now calculate the parameters of the equivalent node. We find

$$\mu' = \mu_n = \mu + \frac{\mu}{n+1}$$

$$S' = \min_{i=1\ldots n}(S_i) = S_n \Longrightarrow B' = \frac{8.5n(n+3)}{n+1}$$

In figure 4, we plot the throughput of the two models (the one that considers all the nodes 1 to n and the other that replaces them with an equivalent node) for the different values of n.

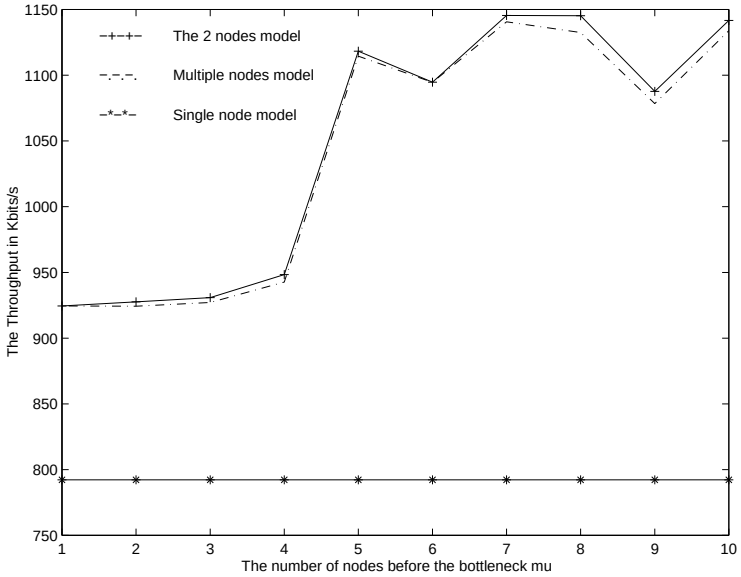

Figure 4: A simulation comparison between the three models

We plot also in the same figure the throughput obtained when we consider only the main bottleneck μ and we feed it directly by the source. Because it is now fed at twice its rate instead of μ_n, the burst size needed to overflow the buffer B passes from $20(n+2)$ to $2B = 40$. This new value of S is always smaller than the minimum of S and S' when the nodes 1 to n are considered whatever is the value of n, of course using our assumptions on B, B_i and μ_i. A smaller S means a smaller window at the beginning of congestion avoidance. Thus, we must expect a throughput that is less than that obtained when the preceding nodes are considered.

We see well that the results are very close for our two models. However, as we expected, the single node model leads to a poorer performance because it doesn't consider the likelihood of a partial absorption of the bursts by the preceding nodes. It represents the worst case where the bursts are only absorbed by the main bottleneck. Also, we see an increase in the throughput when the network contains five nodes and more. Indeed, a large number of nodes on the path increases S_n and S, therefore W_B, which causes a disappearance of the double slow-start phenomena, hence an improvement in the performance. In fact, the bursts during slow-start are absorbed by more and more nodes which eliminates the possibility of the overflow encountered with small values of n.

6 A study of TCP performance using the simplified model

From the previous sections we have

$$S' = \frac{2\mu}{2\mu - \mu'} \times B' = \min_{i=1\ldots n}(S_i) \tag{2}$$

$$S = \frac{\mu'}{\mu' - \mu} \times B \tag{3}$$

$$S_B = min(S, S')$$

$$2W_B = 2^{n_B} + S_B \text{ with } 2^{n_B - 1} < S_B \leq 2^{n_B}$$

In case of $W_B < W_{th} = W_{max}/2 = (\mu T + B)/2$, one of the buffers overflows during slow-start and the loss is detected at a window

$$W_D = min(2W_B, W_{th})$$

This overflow can happen in either B or B'. Its location depends on the relative values of S and S' which are the size of the burst required to fill B and B' respectively. A second slow-start is then called with a threshold $W'_{th} = W_D/2$. Thus, the window size at the beginning of the congestion avoidance phase (equals to W'_{th}) and, therefore, the performance of TCP are a function of the parameters μ', B', μ and B. If the queuing time is negligible with respect to T (which is the case when operating in networks having small buffers compared to their Bandwidth-Delay product), the congestion window can be supposed to vary linearly as a function of time by one segment every T. Thus, the throughput can be approximated as the ratio of the average window size to T. Given that the bulk data transfer happens during congestion avoidance, the average window size can simply be written as $\overline{W} = \frac{W_S + W_E}{2}$ where W_S and W_E are respectively the window size at the beginning and at the end of the congestion avoidance phase. W_E is equal to W_{max}. However, W_S can either be equal to W'_{th} if a double slow-start appears ($W_B < W_{th}$) or W_{th} otherwise. Therefore,

$$Thrp \simeq \begin{cases} \frac{3(\mu T + B)}{4T} & \text{if } W_B > W_{th} \\ min(\frac{W_B + \mu T + B}{2T}, \frac{5(\mu T + B)}{8T}) & \text{otherwise} \end{cases}$$

It is clear that the minimum value of S and S' over all possible values of μ' satisfying $\mu \leq \mu' \leq 2\mu$, is $2B$ and $2B'$ and it corresponds to $\mu' = 2\mu$ and $\mu' = \mu$ respectively. Any decrease in μ' increases S and decreases S' as shown in Figure 5. Thus, the overflow becomes more likely to happen in B' than in B. If B and B' are chosen so that no losses occurs when $\mu' = \mu$ and when $\mu' = 2\mu$, then the double slow-start cannot appear for any μ' satisfying $\mu \leq \mu' \leq 2\mu$. This requires that

$$\left.\begin{array}{l} \mu T + B < 2^{n_B} + 2B \text{ with } 2^{n_B - 1} < 2B \leq 2^{n_B} \\ \mu T + B < 2^{n'_B} + 2B' \text{ with } 2^{n'_B - 1} < 2B' \leq 2^{n'_B} \end{array}\right\} \tag{4}$$

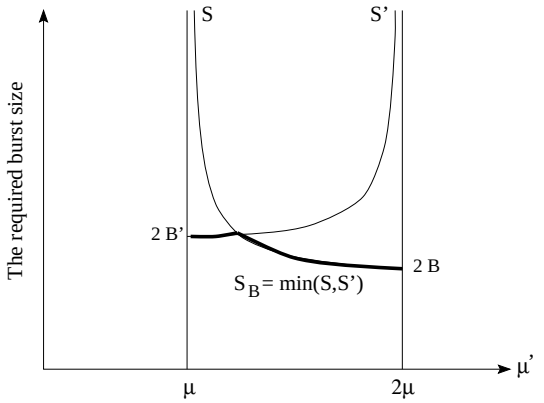

Figure 5: The variation of S and S' as function of μ'

If we approximate 2^{n_B} by $2B$ and $2^{n'_B}$ by $2B'$ as in [11], we get the following conditions on B and B' so as to not have a loss in slow-start whatever is the value of μ'

$$B > \frac{\mu T}{3} \text{ and } B' > \frac{\mu T + B}{4}$$

The shaded region in figure 6 represents the possible values of B and B' for a given μT. It is clear that any increase in B beyond $\mu T/3$ to improve W_{max}, therefore the network resources utilization[3], doesn't require an increase in the buffering capacity in node μ' by the same amount. If we take as an example the case of $B = \mu T$, one half the pipe size is enough as buffer size in node μ' to absorb completely the bursty traffic sent during slow-start whatever is the value of μ'.

In real networks, the intermediate rate μ' may be a variable quantity function of the cross traffic. S_B may then be equal to one of the two minimum values of S and S' ($2B$ and $2B'$), thus any dimensioning of buffers B and B' must satisfy equation (4). Because any node preceding the bottleneck μ is a candidate to be our equivalent node μ', the buffering capacity in these nodes must not be less than the B' given by equation (4)[4]. In this case, we are sure that the slow-start traffic cannot create an overflow neither in node μ nor in the nodes preceding it.

Now, if the problem appears at $\mu' = 2\mu$ (resp. at $\mu' = \mu$), it can be avoided by decreasing (resp. increasing) μ' if the buffer B' (resp. B) is large enough. Thus, a change in μ' can improve TCP performance without needing to increase the capacity of the buffers.

[3]To obtain a 100% link utilization during congestion avoidance, the slow-start threshold W_{th} must be at least equal to the pipe size. This requires a buffer B larger than μT.

[4]By buffer size we mean the share of TCP packets in the global buffer in a node.

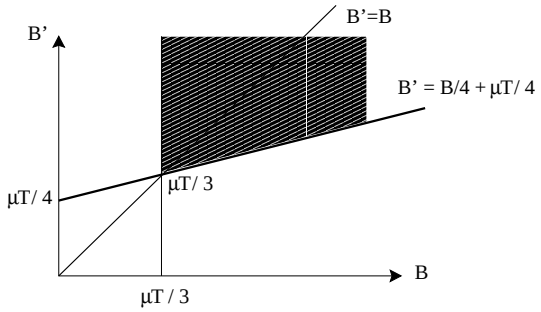

Figure 6: The required B' as a function of B

In fact, a μ' between μ and 2μ spreads the queue (and thus the congestion) over the two buffers reducing the likelihood of an overflow during slow-start. Again, if we approximate 2^{n_B} by S_B, we get $W_B = S_B$, thus the appropriate value for μ' is the one that gives an S_B (given by the thick line in figure 6) larger than $W_{th} = (\mu T + B)/2$.

In figure 7, we see simulation results showing how the throughput varies as a function of μ'. We take the network parameters so that a double slow-start appears in B when $\mu' = 2\mu$. For a large B' (80 packets), the throughput jumps up at a certain $\mu' < 2\mu$ (as we decrease μ') due to the disappearance of the double slow-start but it doesn't decrease if we further decrease μ'. This is because B' is large enough so it is able to absorb alone the bursty traffic (when $\mu' = \mu$).

For a medium B' (45 packets), the throughput increases as we decrease μ' only in some intermediate range where the bursty traffic is spread over the two buffers. If we further decrease μ', it then decreases too due to the reappearance of the double slow-start phenomena.

However, for a small B', the double slow-start isn't solved even in the middle, hence, we don't see the jump in the throughput. Moreover, it deteriorates for small μ' due to the decrease in the window size at the beginning of the congestion avoidance phase (W'_{th}).

The behavior just described may seem quite paradoxical, as decreasing the throughput of a link (μ') results in an increase in the throughput. Note that a similar phenomena was observed in the context of flow control in ATM (ABR services) in [1].

Using the expressions of S and S' in (2) and (3), we notice that their values remain approximately unchanged for most of the interval $[\mu, 2\mu]$ and then they move quickly to infinity. In figure 8, we plot the variation of S as a function of μ'/μ for different values of B (10, 20, 30, 40 and 50). This behavior becomes more evident when the buffer size decreases. This

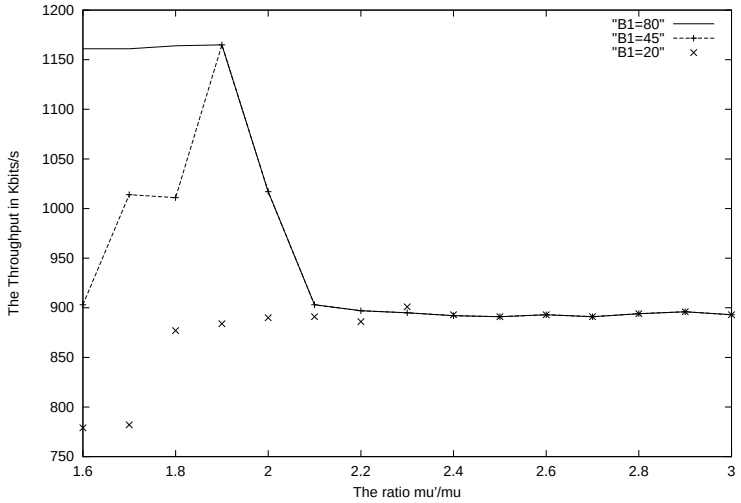

Figure 7: The simulated throughput vs. the intermediate rate μ'

means that if B (resp. B') is so small so that an overflow occurs at $\mu' = 2\mu$ (resp. $\mu' = \mu$), the overflow window W_B remains approximately unchanged and equals to $2B$ (resp. $2B'$) whatever are the intermediate bandwidth μ' and the buffer size in the other node. Thus, we can say that the window at which a loss occurs during slow-start is, for most of the cases, equal to twice the smallest buffer on the path (we use the approximation $2^{n_B} \simeq 2B$).

7 The disappearance of the second slow-start

In this section we study a phenomena discovered while simulating TCP in networks satisfying the condition to get the double slow-start problem. In some scenarios, the retransmission of the packet lost during the first slow-start of a TCP cycle is systematically lost resulting in a disappearance of the second slow-start. Indeed, when the loss in the first slow-start is detected, the source sets its window to 1, its threshold to W'_{th} and then calls a new slow-start by retransmitting the lost packet. A loss of this retransmission forces the source to resort to a long timeout to detect it. Again, at the end of this timeout, it sets its threshold to half its current window, it initializes its window to one and it calls a new slow-start. But the congestion window was previously one, therefore the threshold is set to its minimum possible value 2. With this low slow-start threshold, a congestion avoidance phase starts very early which results in a considerable throughput deterioration. It takes the source a long time to fully utilize the available capacity. Using the linear throughput approximation given in the

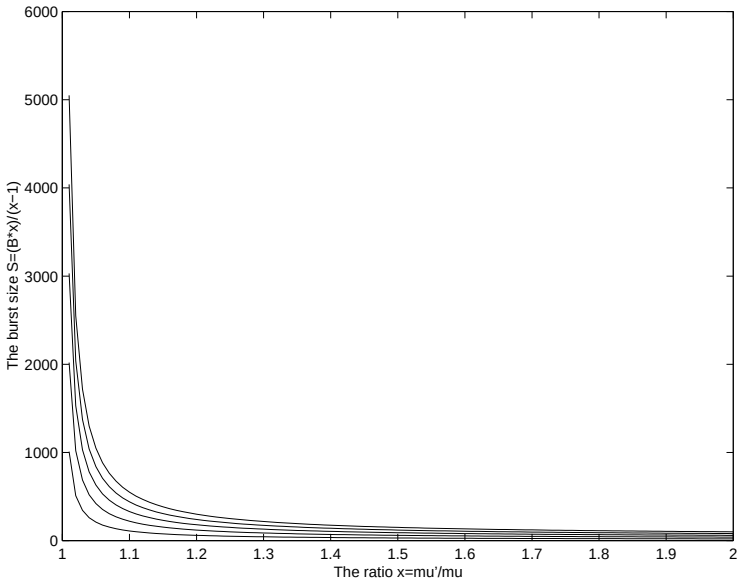

Figure 8: The burst size S vs. the intermediate rate μ'

previous section, this problem reduces the average window size of the TCP connection from $\frac{W'_{th}+W_{max}}{2}$ to $\frac{W_{max}}{2}$. This deterioration is maximum when $W'_{th} = W_{max}/4$ and is equal to $1 - 1/(1 + 0.25) = 0.2$. Thus, TCP can lose 20% of its average window, and therefore of its throughput, due to this undesirable loss. To characterize this problem, we proceed as follows.

We consider first the case of the multiple node model given in figure 1. The source detects the loss after the receipt of three duplicate ACKs. This detection happens in mini-cycle $n_B + 1$ which follows the mini-cycle where the loss occured. Between the beginning of $n_B + 1$ and the receipt of the first duplicate ACK, the source sends some packets of number Q which may overflow some of the network buffers. If during the time needed to detect the loss, the buffer in any bottleneck is maintained full, the retransmission may not find a place in this buffer and consequently it might be lost. The number of new ACKs (an ACK carrying a new information) received during $n_B + 1$ before the loss detection is simply equal to S_B. These ACKs arrive at rate μ. Let D be the time elapsed between the beginning of this mini-cycle and the loss detection, therefore

$$D = \frac{S_B + 3}{\mu}$$

To maintain a buffer B_i full until the arrival of the retransmission, Q must be large enough so that to be able to feed this buffer at rate μ_{i-1} during time D. This can be expressed by

$$Q > D \times \mu_{i-1}$$

Also, there is another condition. Feeding the buffer B_i at rate μ_{i-1} during D must overflow it. Thus, the burst size S_i must be smaller than the number of packets injected into the buffer. We get thus the second condition

$$S_i < D \times \mu_{i-1}$$

As a result, the retransmission might be lost if on the path we find one bottleneck node that satisfies

$$S_i < D \times \mu_{i-1} < Q \tag{5}$$

To compute Q, and then to understand the problem, we must consider the relative position of W_{th} with respect to W_B and $2W_B$. Given that 2^{n_B} is the size of the window at the beginning of mini-cycle $n_B + 1$, we can write

$$\begin{array}{ll} Q = S_B + W_{th} - 2^{n_B} = 2S_B + W_{th} - 2W_B & \text{if } W_B < W_{th} \leq 2W_B \\ Q = 2S_B & \text{if } W_{th} > 2W_B \end{array}$$

These two expressions of Q can be summarized by a single one

$$Q = 2S_B + min(0, W_{th} - 2W_B)$$

The location of the retransmission loss is the nearest bottleneck to the source that satisfies the condition (5). Note here that this loss cannot happen in node 1. In general, it cannot happen in a node fed directly by the source at a rate faster than 2μ. Such access link lets the source send a burst of two packets before the arrival of the next ACK. Thus, the retransmission is sent directly upon the detection of the loss and the time elapsed during the receipt of the duplicate ACKs, which is equal to $3/\mu$, guarantees that at least one place is freed in the buffer of the first bottleneck so that the retransmission will not be lost. A network with an access link μ_A slower than 2μ is a particular case of our model if we take $\mu_0 = \mu_A$ upon the calculation of S_1 using equation (1). In this case, a loss of the retransmission can occur in all the nodes of our model.

Thus, the necessary condition to get this phenomena is

$\exists i$ ($i \neq 1$ if $\mu_A = \mu_0 > 2\mu$) such that

$$S_i < D \times \mu_{i-1} < 2S_B + min(0, W_{th} - 2W_B)$$

with $S_{n+1} = S$.

It is clear here that if the source gets in congestion avoidance before the beginning of mini-cycle n_B+1 which is represented by $W_{th} < 2^{n_B}$, Q becomes smaller than S_B. Given that S_B is the minimum of all the S_i and S, the previous condition cannot be satisfied for any node in this case. Therefore, the slow-start may disappear only if the source enters the mini-cycle which follows the overflow and she is still in slow-start.

In our simplified model with two bottlenecks, the loss of the retransmission can only appear in buffer B, of course if the source access link is faster than 2μ. We must thus examine the condition only at one node. We get

$$S < D \times \mu' < 2S_B + min(0, W_{th} - 2W_B)$$

Here, S_B, and then D, can take one of two values depending on in which node the loss in the first slow-start has happened. If $S < S'$, $S_B = S$ and the condition $S < D \times \mu'$ is then satisfied. Only the other condition $D \times \mu' < Q$ must be examined. In the other case, the two conditions must be simultaneously examined.

Consider now the particular case of a source connected to the network via a link of rate $\mu_A = \mu' < 2\mu$. Suppose also that the there is no slow links between the source and the main bottleneck. Such network can be modeled by a single bottleneck node of rate μ and a slow source access link μ'. In this case, the loss of the retransmission may appear in node μ. We have $S_B = S$ and the necessary condition to get this loss is

$$D \times \mu' = (S+3)\frac{\mu'}{\mu} < 2S + min(0, W_{th} - 2W_B)$$

If we ignore the 3 in the left side in front of S and we use the approximation $W_B \simeq S$, we find the following condition to see the loss of the retransmission which is always true for $W_{th} > 2W_B$

$$W_{th} > S \times \frac{\mu'}{\mu} = W_B \times \frac{\mu'}{\mu} \Longrightarrow \beta < \frac{1}{4\frac{\mu'}{\mu} - 1}$$

with $1 < \frac{\mu'}{\mu} < 2$. As we have said before, we cannot see it for a μ' very close to 2μ even if this condition is satisfied.

7.1 The case of RED buffers

This disappearance of the second slow-start is first due to a fast retransmission of a packet before giving the network enough time to serve the bursts we have already sent. Also, it is the result of the Drop Tail policy in the gateways which drop the incoming packet when the buffer gets full. Using other buffer management strategies can form a solution without requiring a change in TCP behavior. The idea is to reduce the drop probability of a packet that arrives to a full buffer and to replace it by another one already in the queue. Because the first retransmission after loss detection is followed by a silence period at the source (of

one RTT in case of Tahoe), giving the retransmission the place of another packet in the queue when it finds the buffer full increases significantly its probability to be successfully received.

Random Early Detection (RED) [7] gateways provide a class of policies that drop a packet when the average queue length exceeds a certain threshold. The packet to drop can be the last incoming one or another packet chosen randomly from those waiting in the queue. Even if RED is used, choosing the last incoming packet to be dropped can keep the same problem of the retransmission loss. On the other hand, the random choice avoids the problem by giving priority to the new packet and thus to the retransmission. However, avoiding a retransmission loss doesn't mean necessary that these gateways give always better performance than the Drop Tail ones. By dropping another packet than the last incoming one, they change the behavior of the connection every time the buffer gets full not only when the retransmission arrives. There is no way to distinguish between a retransmission and a packet sent for the first time. This may result in a different window size at the beginning and at the end of the congestion avoidance phase and thus in a different performance. The effect of these gateways on the performance is studied by simulations in the next section.

Remark 1 *A possible solution to the problem at the source level, that doesn't require any change in the network, might be to wait a certain time before the retransmission or to retransmit directly the lost packet more than once. This action must be taken by the source only when it transmits the first packet after the detection of a loss. Waiting a certain time may not avoid completely the problem. Even if it is delayed, the retransmission may encounter a non empty buffer feeding a full buffer downstream. Here, it may be lost again. On the other hand, duplicating the lost packet many times ensures that if a retransmission is lost in some queue, the other retransmitted packets cannot be lost in the same queue. However, the effect of such solution on the overall performance must be investigated.*

7.2 Simulations

The scenario that interests us is one that satisfies the condition for a loss of the retransmission on more than one node. We take a network with two bottleneck nodes of rates $\mu = 1.5$ Mbits/s and $\mu' = 2.2$ Mbits/s and a slow source access link of rate 2.7 Mbits/s. To obtain many results, we set B' to 15 packets and we vary B between 10 and 20 packets. The packet size and T are chosen as before.

To study the efficiency of a solution based on a buffer management strategy different that Drop Tail, we simulate two simple gateways that can be considered as particular cases of RED. Random Drop (RD) gateways [8] drops randomly a packet from the queue when the buffer gets full. The other gateway called Drop from Front (DF) strategy [12] proceeds in the same manner but chooses always the packet at the front of the queue to be dropped. For $B = 18$ packets, we see in figure 9 how the loss of the retransmission makes the second slow-start disappear. Supplying the two nodes with RD buffers results in the reappearance

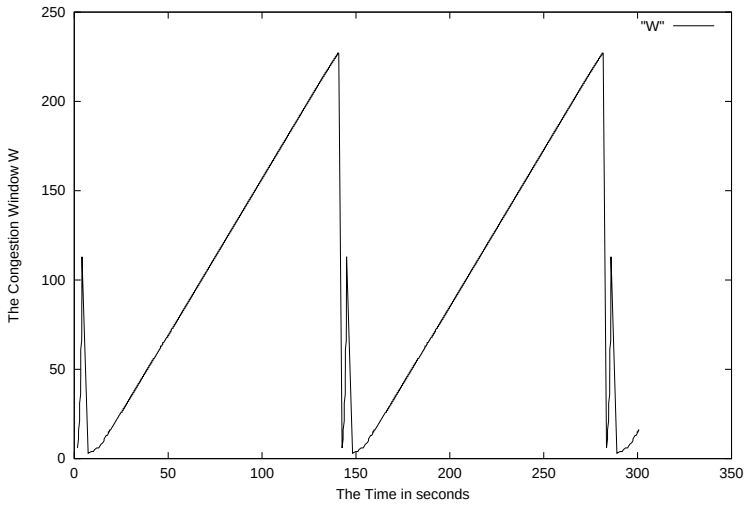

Figure 9: The window evolution with Drop Tail buffers

of the second slow-start and thus in the improvement of the throughput from 717 Kbits/s to 829 Kbits/s as shown in figure 10.

Figure 11 compares TCP throughput when the problem of the retransmission loss exists due to Drop Tail buffers to that obtained when the two gateways DF and RD are deployed in the network. We remark first that the throughput in case of Drop Tail buffers oscillates due to the appearance and disappearance of the problem. We notice also that when the problem exists (B=13, 16 and 18), RD and DF gateways avoid a loss of the retransmission and increase the throughput. However, their side effect on the behavior of the connection impairs the performance of TCP when the problem doesn't exist. Indeed, dropping a packet already in the queue (during mini-cycle n_B of the first slow-start phase) makes the source detect quickly the buffer overflow during slow-start instead of waiting the receipt of all the packets in the queue (of number S_B). This earlier detection is bad because it reduces the window size when the loss is detected W_D. This can be interpreted by the receipt of less than S_B ACKs during mini-cycle $n_B + 1$. The result is a smaller window at the beginning of the congestion avoidance phase (equals to $W_D/2$ when the problem of retransmission loss doesn't exist) and thus a decrease in the throughput. This behavior cannot occur at the end of congestion avoidance where the window change slowly by one segment for every W ACKs. Thus, an earlier detection of the loss doesn't change W_{max} which keeps its old value $B + \mu T$. This decrease in the number of ACK received during mini-cycle $n_B + 1$ and thus in W_D is in its worst case with the DF gateway. The number of these ACKs is equal to $S_B/2$ versus S_B when Drop Tail is used. In case of RD gateway, this number is somewhere

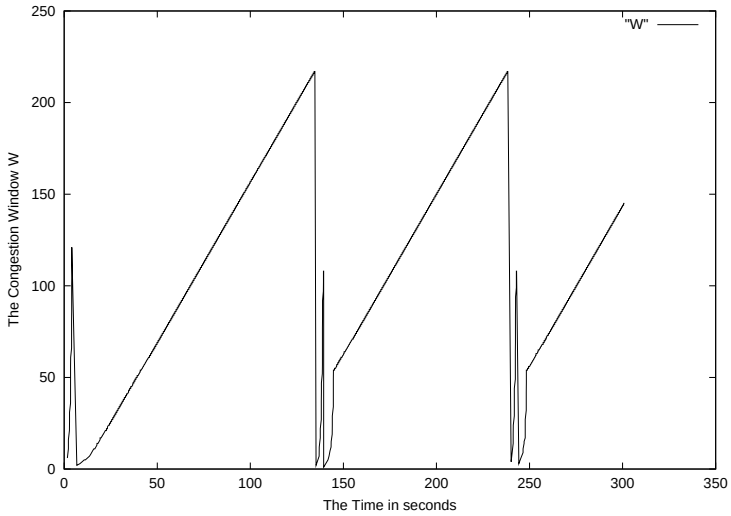

Figure 10: The window evolution with Random Drop buffers

between the two. Given this number, we can calculate the general expression of the window size at the beginning of congestion avoidance, of course if a buffer overflows during slow-start

$$W_S = W_D/2 = min(W_{th}, 2W_B + (\alpha - 1)S_B/2)$$

where $\alpha = 1$ for a Drop Tail buffer, $\alpha = 0$ for a DF one and α equals to a random number between 0 and 1 for a RD buffer.

Because W_{max} remains the same, the comparison between the throughput obtained with the three policies must be based on W_S (using the approximation of the throughput given in section 6). This window is always greater than 2, therefore the two simple gateways give better performance when the Drop Tail one causes a loss of the retransmission. However, RD gives a window W_S larger than that in the DF case which explains the difference between the two throughputs in favor of RD for certain B (13 and 20). Because this window increases with α, the Drop Tail gateway gives the best performance when the retransmission is not lost. We must mention here that this conclusion is only true when the slow-start causes an overflow in one buffer (the case $W_B < W_{th}$). If this doesn't happen, W_S will be equal to half W_{max} which is not a function of the factor α, thus we get the same performance with the three drop policies.

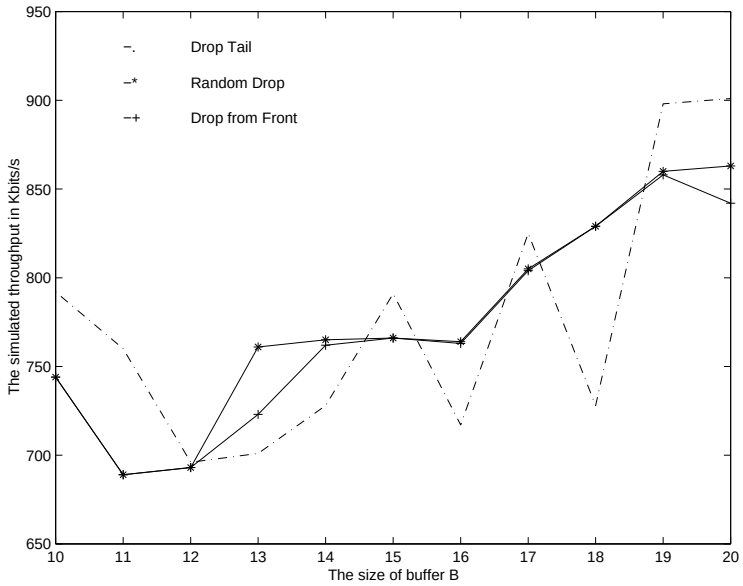

Figure 11: The effect of the buffer management policy on the throughput

8 Conclusion

In this paper, we studied the performance of TCP as a function of the parameters of all the nodes on the path between the source and the destination. We presented a general model taking into account all these nodes not only the one with the slowest outgoing rate called the main bottleneck in many works. Thanks to this model, we showed that the behavior of TCP can change compared to that predicted by a simple model consisting only of the main bottleneck. This change is minor during congestion avoidance but it may lead to a completely different slow-start phase. The bursts sent during slow-start might be absorbed by some nodes between the source and the main bottleneck. This partial absorption can avoid an overflow during slow-start predicted by the single node model which improves considerably the throughput of the connection. But, it can also cause an overflow in another node if its buffer is not well dimensioned even if the buffer in the main bottleneck is set too large. It is known that an overflow during slow-start yields an underestimation of the available capacity in the network and therefore in a deterioration of the throughput. Then, we prove that we can simplify the analysis without changing the performance by modeling the network with two bottleneck nodes. One node represents the main bottleneck and the other represents all the other nodes on the path. We gave the expressions of the parameters of this model as a function of network parameters. Using this simplified model, we studied the performance of TCP and we presented guidelines for the dimensioning of network buffers

in order to get a better throughput. The buffer in the main bottleneck determines alone the behavior of TCP during congestion avoidance and it must be increased to improve the resources utilization. Although they don't affect the behavior during congestion avoidance, the buffers in the other nodes are however necessary to absorb a potential congestion due to slow-start bursts. These buffers must scale linearly with that in the main bottleneck but not necessary in the same amount.

Finally, we presented a problem discovered while simulating TCP in networks with small buffers compared to their bandwidth-delay product. The packet retransmitted after the detection of a loss may be systematically lost. This results in a long timeout and a disappearance of the slow-start phase hence in a deterioration of the throughput. Because it is a result of the Drop Tail policy in the buffers, we studied by simulations the behavior of the connection when RED gateways with a random drop policy are deployed in the Internet. We showed that these gateways resolve the problem by dropping another packet in the queue than the retransmission. However, because they don't distinguish between packets sent for the first time and retransmissions, these gateways may deteriorate the throughput if the problem doesn't exist. Indeed, a drop from inside the queue reduces the window size at the beginning of the congestion avoidance phase, and therefore the average window size, when the first slow-start in a TCP cycle overflows the network buffers.

References

[1] O. Ait-Hellal and E. Altman, "Performance Evaluation of Congestion Phenomena in the Rate Based Flow Control Mechanism for ABR", in proceedings of IEEE INFOCOM 99, New York, March 1999.

[2] E. Altman, J. Bolot, P. Nain, D. Elouadghiri- M. Erramdani, P. Brown, and D. Collange, "Performance Modeling of TCP/IP in a Wide-Area Network", INRIA-France, Research Report N=3142 (1997) (available at http://www.inria.fr:80/RRRT/RR-3142.html). A shorter version in: *34th IEEE Conference on Decision and Control*, December 1995, New Orleans, Louisiana, (invited paper), pp. 368-373.

[3] Ch. Barakat and E. Altman, "Analysis of TCP in Networks with Small Buffering Capacity and Large Bandwidth-Delay Product", Research Report N=3574 (1998) (available at http://www.inria.fr:80/RRRT/RR-3574.html).

[4] N. Chaher, Ch. Barakat, W. Dabbous, and E. Altman, "Improving TCP/IP over Geostationary Satellite Links", Research Report N=3573 (1998) (available at http://www.inria.fr:80/RRRT/RR-3573.html).

[5] H. Chaskar, T. V. Lakshman, U. Madhow, "On the design of interfaces for TCP/IP over wireless", invited paper, in Proceedings of IEEE Milcom '96.

[6] K. Fall and S. Floyd, "Simulation-based Comparisons of Tahoe, Reno, and SACK TCP", Computer Communication Review, V. 26 N. 3, July 1996, pp. 5-21.

[7] S. Floyd and V. Jacobson, "Random Early Detection gateways for Congestion Avoidance", IEEE/ACM Transactions on Networking, 1(4):397-413, August 1993.

[8] E. Hashem, "Analysis of random drop for gateway congestion control", Report LCS TR-465, Laboratory for Computer Science, MIT, Cambridge, MA, 1989, p.103.

[9] V. Jacobson, "Congestion avoidance and control", in Proceedings of ACM Sigcomm'88, Stanford, CA, USA, Aug. 1988.

[10] A. Kumar, "Comparative Performance Analysis of Versions of TCP in a Local Network with a Lossy Link", IEEE/ACM Transactions on Networking, August 1998.

[11] 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, pp. 336-350, June 1997.

[12] 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", in Proceedings of Infocom'96, pp. 1242-1250.

[13] S. McCanne and S. Floyd, NS (Network Simulator), 1995, URLs http://www-nrg.ee.lbl.gov/ns, http://www-mash.cs.berkeley.edu/ns.

[14] W. Stevens, "TCP Slow-Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms", RFC 2001 (1997).

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles - B.P. 93 - 06902 Sophia Antipolis Cedex (France)

Unité de recherche INRIA Lorraine : Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - B.P. 101 - 54602 Villers lès Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38330 Montbonnot St Martin (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - B.P. 105 - 78153 Le Chesnay Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, B.P. 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399