A note on the Fairness of TCP Vegas

Catherine Boutremans and Jean-Yves Le Boudec

Institute for Computer Communication and Applications (ICA)
Ecole Polytechnique Fédérale de Lausanne(EPFL)
1015 Lausanne, Switzerland
[Catherine.Boutremans,Jean-Yves.Leboudec]@epfl.ch

Abstract

We study the fairness of TCP Vegas. The latter is an alternative to the commonly used TCP Reno, and uses measures of the round trip time as feedback on congestion. We consider two cases that depend on the value of the two parameters α and β controlling the window sizes' update.

Our main conclusion is that TCP Vegas is unfair in several points. First, when $\alpha = \beta$, if the propagation delays are correctly estimated, TCP Vegas is known to be fair. However we show that any over-estimation of the propagation delay of a given connection results in an increase of its rate and hence leads to unfairness. This rate increase augments with the over-estimation factor. Moreover, the rate oscillations, whose amplitude increases with the rate value, are not sufficient to provide an accurate estimation of the propagation delay. Second, when $\alpha < \beta$, TCP Vegas is unfair even if the propagation delays are correctly estimated. In this case, the rate of a connection converges to a stable value that depends on the arrival order of all connections so that earliest established connections get more bandwidth. Also, in a more realistic scheme, later connections see their propagation delay over-estimated and thus they gain larger portion of the bandwidth. These two effects tend to counterbalance each other but the second tends to dominate.

Future use of TCP Vegas in the context of TCP-friendly applications, should therefore rely on $\alpha = \beta$, but will require the propagation delays to be correctly estimated. Yet, this seems to be quite hard to achieve.

1 Introduction

Our objective in this paper is to better understand the dynamics of the congestion control algorithm implemented in TCP Vegas [3] and to better assert its possible use in the context of TCP friendly applications.

First, we briefly introduce the principles of TCP Vegas, detailed in Section 2, which uses measures of round trip time (RTT) as congestion feedback, rather than packet losses. In Vegas, actual and expected rates in a connection are evaluated using, respectively, the actual value of the RTT and the minimum value of all RTT values ever measured in this connection (this is the estimation of the propagation delay). The difference between these rates, whose value is compared to two parameters (namely α and β), is then used to adjust the window size.

Several drawbacks of TCP Vegas have been pointed out recently. First, because of the default values of α and β in its implementation [3, 1], TCP Vegas does not share the total bandwidth amongst connections in a fair way [2, 6]. Secondly, the same unfairness is observed for any given inaccurate estimation of the propagation delay.

This led Hasegawa [5, 4] to propose an enhanced Vegas in which $\alpha = \beta$. He shows that this setting leads to oscillations in the window size values, and claims that their amplitude can provide an accurate estimate of the propagation delay. This is crucial as even in this setting, the accurate estimate is necessary to ensure a fair share of the bandwidth.

In this paper, we first check the fairness of the enhanced TCP Vegas proposed by Hasegawa. In particular, we address the problem of the propagation delay estimation. This leads us to review the case where $\alpha < \beta$. Our main results follow.

Under the $\alpha = \beta$ setting, we find that the rate oscillations, which we show increase with the rate value, do not allow a connection to accurately estimate the propagation delay. Also, any over-estimation of the propagation delay of a given connection will increase its rate, an increase that becomes more pronounced with the over-estimation factor. In return, the different connections' peaks in the oscillations are not synchronized in time, thus there is no sub-use of the link capacity.

When $\alpha < \beta$, we show that the rate of a connection converges to a stable value that depends on the arrival order of all connections. As a result, the first connections to be established will be favored when the propagation delays are properly estimated. Yet, in later connections the propagation delays are overestimated and so their rates are greater than what they should be. These two effects tend to counterbalance each other but the second tends to dominate.

This paper is organized as follows. In section 2, we summarize the congestion avoidance scheme of TCP Vegas. Section 3 gives an analysis of the case $\alpha = \beta$ and presents simulation results. Section 4 is devoted to the case $\alpha < \beta$. Finally, a discussion on our results is addressed and conclusions are drawn.

2 TCP Vegas' Congestion Control Algorithm

In this section, we describe the congestion avoidance algorithm of TCP Vegas. As mentioned previously, the bandwidth estimation scheme of TCP Vegas radically differs from the one of TCP Reno. While TCP Reno uses packet losses as congestion feedback, TCP Vegas uses the difference between the expected and actual rates to estimate the congestion state of the network. Because TCP Vegas does not need to engender losses to evaluate the available bandwidth in the network, it utilizes the bandwidth more efficiently than TCP Reno.

The basic idea of TCP Vegas is that the farther away the actual throughput gets from the expected throughput, the more congested is the network, which implies that the sending rate should be reduced. The threshold β triggers this decrease. On the other hand, when the actual and expected throughputs are close, the connection is in danger of not utilizing the available bandwidth. The threshold α triggers this increase.

The Congestion Avoidance algorithm of TCP Vegas, first introduced in [3], can be summarized as follows. Once per round trip time,

1. Vegas computes the expected throughput, which is given by:

$$Expected = cwnd/baseRTT$$

   where $cwnd$ is the current window size and $baseRTT$ is the minimum of all measured round trip times.

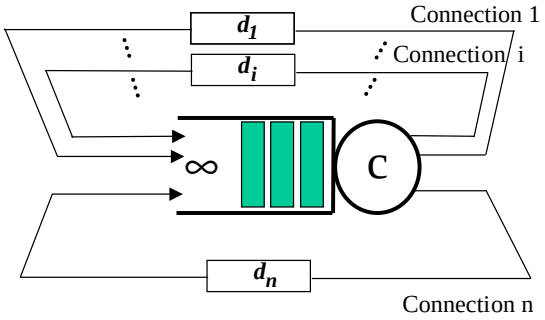

Figure 1: *Network model*

2. Vegas calculates the current *Actual* sending rate by using the actual round trip time:

$$Actual = cwnd/RTT$$

where RTT is the observed round trip time of a packet.

3. Vegas computes the estimated backlog in the buffers by:

$$Diff = (Expected - Actual) * baseRTT$$

4. finally, Vegas updates the window size as follows:

$$cwnd = \begin{cases} cwnd + 1 & if\ Diff < \alpha \\ cwnd & if\ \alpha \leq Diff \leq \beta \\ cwnd - 1 & if\ Diff > \beta \end{cases} \quad (1)$$

TCP Vegas controls its window size to keep the measured backlog within the boundaries $[\alpha..\beta]$. The reason behind is that TCP Vegas tries to detect and utilize the extra bandwidth whenever it becomes available without congesting the network. Typical values of α and β are 1 and 3 or 2 and 4 [3, 1] .

3 Case 1: $\alpha = \beta$

As we have just seen, TCP Vegas tries to keep a certain amount of packets queued in the buffers. This implies that the value of $baseRTT$ can be greater than the propagation delay (which is the delay when there is no queue). In this section, in which $\alpha = \beta$, we will analyze the influence of an over-estimation of the propagation delay on the fairness of TCP Vegas.

3.1 Analysis

3.1.1 Analytical study of the steady state

In this analysis of the fairness of TCP Vegas, we propose a generalization of the equations presented in [6]. We will study the rate distribution, provided by TCP Vegas, at the *steady state* that is when all the window sizes have converged to a stable value.

The network model considered here is illustrated in Figure 1. It consists of a single bottleneck link, shared by n users. User i $(i = 1, ..., n)$ has a propagation delay d_i and uses the window-based flow control of TCP Vegas. The bandwidth of the link is c and the switch adopts

a FIFO discipline. The buffer size is assumed to be infinite. This assumption ensures that TCP Vegas does not behave like TCP Reno, which would be the case for small buffer sizes (as shown in [2]).

In the depicted configuration, let us assume that each user i measures a minimum round trip time,

$$baseRTT_i = d_i + x_i \tag{2}$$

where x_i (≥ 0) is the propagation delay over-estimation of connection i.

We now consider that the TCP Vegas algorithm has reached a steady state (fixed window sizes). Then, each connection i measures a round trip time $RTT_i = d_i + \Delta$ where Δ is the queuing delay at the switch.

We can deduce from (1) that, at the steady state, $Diff = \alpha$. Therefore, we can express the window sizes by:

$$cwnd_i - \frac{baseRTT_i}{RTT_i}\, cwnd_i = \alpha$$

or

$$cwnd_i - \frac{d_i + x_i}{d_i + \Delta}\, cwnd_i = \alpha \tag{3}$$

$$cwnd_i = \alpha\, \frac{d_i + \Delta}{\Delta - x_i} \tag{4}$$

Let us now derive an expression for the throughput of connection i:

$$rate_i = \frac{cwnd_i}{RTT_i} = \frac{\alpha}{\Delta - x_i} \tag{5}$$

This equation clearly shows that any over-estimation x_i of the propagation delay of a given connection results in an increase of its rate which gets greater as the over-estimation factor gets close to Δ .

And finally, if the network capacity c is fully utilized, i.e. $\sum_{i=1}^{n} rate_i = c$, we can deduce the queuing delay at the steady state using:

$$\sum_{i=1}^{n} \frac{\alpha}{\Delta - x_i} = c \tag{6}$$

3.1.2 Particular cases

Here we solve equation (6) for two cases of interest:

- **if all connections measure accurately the propagation delay**, $x_i = 0$ for all i. Then, the solution of (6) is $\Delta = \frac{c}{n\ \alpha}$ and

  $$rate_i = \frac{c}{n}$$

  This case leads to a fair share of the link bandwidth, which confirms the results in [5, 2].

- **if connection i starts when connections $1, ..., i-1$ are in equilibrium**, the value of x_i ($i = 1, ..., n$) has been determined by Bonald in [2]. He showed that $baseRTT_i$, which is

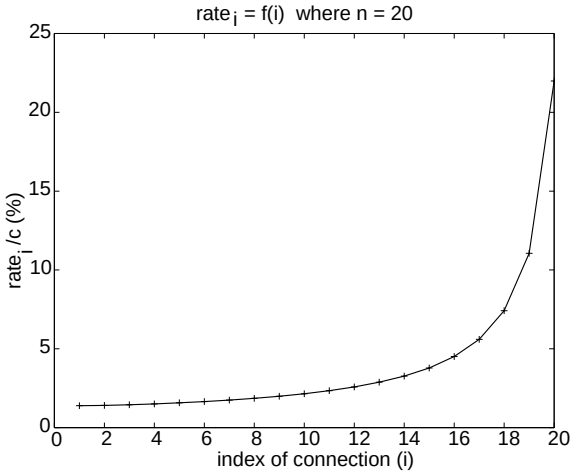

Figure 2: *Repartition of the rates for n = 20*

the measure of the round trip time in the steady state reached by connections $1, ..., i-1$ was given by

$$baseRTT_i = d_i + \frac{\alpha}{c}\ (i-1)\ S_{i-1}$$

where $S_0 = 0$ and for all $i \geq 1$, $S_i = 1 + \frac{1}{2} + \frac{1}{3} + ... + \frac{1}{i}$.
Therefore the solution of (6) is $\Delta = \frac{\alpha}{a}\ n\ S_n$ and

$$rate_i = \frac{c}{n\ S_n - (i-1)\ S_{i-1}}$$

for $i = 1, ..., n$.

Figure 2 shows the repartition of the rates for $n = 20$. The last connection to be established gets 10 times more bandwidth than the earlier connections. This confirms the critical influence of the propagation delay over-estimation on the fairness.

3.1.3 Quantification of the influence of the propagation delay over-estimation

In order to quantify the influence of the propagation delay over-estimation on the rate distribution we computed two partial derivatives of interest:

- the partial derivative of the rate of a connection with respect to its over-estimation factor is

$$\frac{\frac{\partial rate_i}{\partial x_i}}{rate_i} = \frac{\sum_{k \neq i} \frac{1}{(\Delta - x_k)^2}}{1 + (\Delta - x_i)^2 \sum_{k \neq i} \frac{1}{(\Delta - x_k)^2}} (\Delta - x_i) \tag{7}$$

  This derivative has large values when x_i is close to Δ. This shows that the influence of the over-estimation of the propagation delay of a connection on its rate increases with the over-estimation factor.

- the partial derivative of the rate of a connection with respect to the over-estimation factor of another connection is

$$\frac{\frac{\partial rate_i}{\partial x_k}}{rate_i} = -\frac{1}{(\Delta - x_i) + \sum_{j \neq k} \frac{(\Delta - x_k)^2 (\Delta - x_i)}{(\Delta - x_j)^2}} \tag{8}$$

This quantity (which is negative) has significant values only when x_i and x_k are close to Δ and $x_i < x_k$. In practice, this means that the cross influence of the over-estimation of a connection on the rate of another connection is important only for connections with an important over-shooting.

So far, we have considered that all window sizes did stabilize. In fact, when $\alpha = \beta$, the window sizes will oscillate around the steady state values considered in this section. An analytical study of the system dynamics is quite complex. Therefore, we performed simulations to study these oscillations and to check if Hasegawa's hypothesis was true.

3.2 Simulations

The results presented in this section were obtained with two different simulators: the Network Simulator (ns) developed at Lawrence Berkeley Laboratory and our own implementation of the congestion avoidance algorithm of TCP Vegas, to cross-check our results.

3.2.1 Simulation setup

The simple network model that was simulated is the one described in section 3.1. It consists of a single bottleneck shared by n connections (see Figure 1). The following parameters were used: the link bandwidth $c = 1$ Mbps, the propagation delays $d_i = d = 0.2$ s for all i (all users have the same propagation delay), the number of users $n = 10$, 20 and 40, and the buffer size is infinite. The successive connections (i = 1,...,n) join the network every 2 seconds, starting from connection with index 1. In addition, we introduced a random part to the propagation delay in order to take into account the influence of very small variations of the queue size (the random part was a zero mean gaussian with variance equal to the variance of an M/M/1 queue loaded at 90%). Each simulation lasted for 120 seconds.

3.2.2 Results

In this section, we present some results that exhibit the behaviour of the TCP Vegas congestion avoidance phase.

In Figure 3 (a,b,c), we show the values of $baseRTT$ measured by the different connections. Each figure corresponds to a different value of the α parameter. The x-axis and y-axis represent respectively the index of the connection (recall connections join successively the network) and the corresponding $baseRTT$. The solid line represent the theoretical values of $baseRTT$ given by Bonald (not taking the oscillations into account). The triangles and the stars represent respectively the values of $baseRTT$ at the beginning and at the end of the connection. Their simulation values are very close to each other and are far from the value of the propagation delay (especially for the late connections). This means that *the oscillations are not sufficient to allow the connections to measure accurately the propagation delay.* However, the theoretical value of $baseRTT$ is pessimistic compared to the real values.

Let us now investigate the influence of the propagation delay over-estimations on the rates distribution. Figure 4 (a,b,c) shows the rates of the different connections ($rate_i$) as a function of their over-estimation factor (x_i) for different values of α and n. The vertical bars represent the amplitude of the rate oscillations. We can see that any over-estimation of the propagation delay of a connection results in an increase of its rate which gets worse (for a given α) when the over-estimation factor increases. This effect is very critical as the late connections can receive up to 5 times more bandwidth than the earlier connections. This demonstrates the unfairness of TCP Vegas.

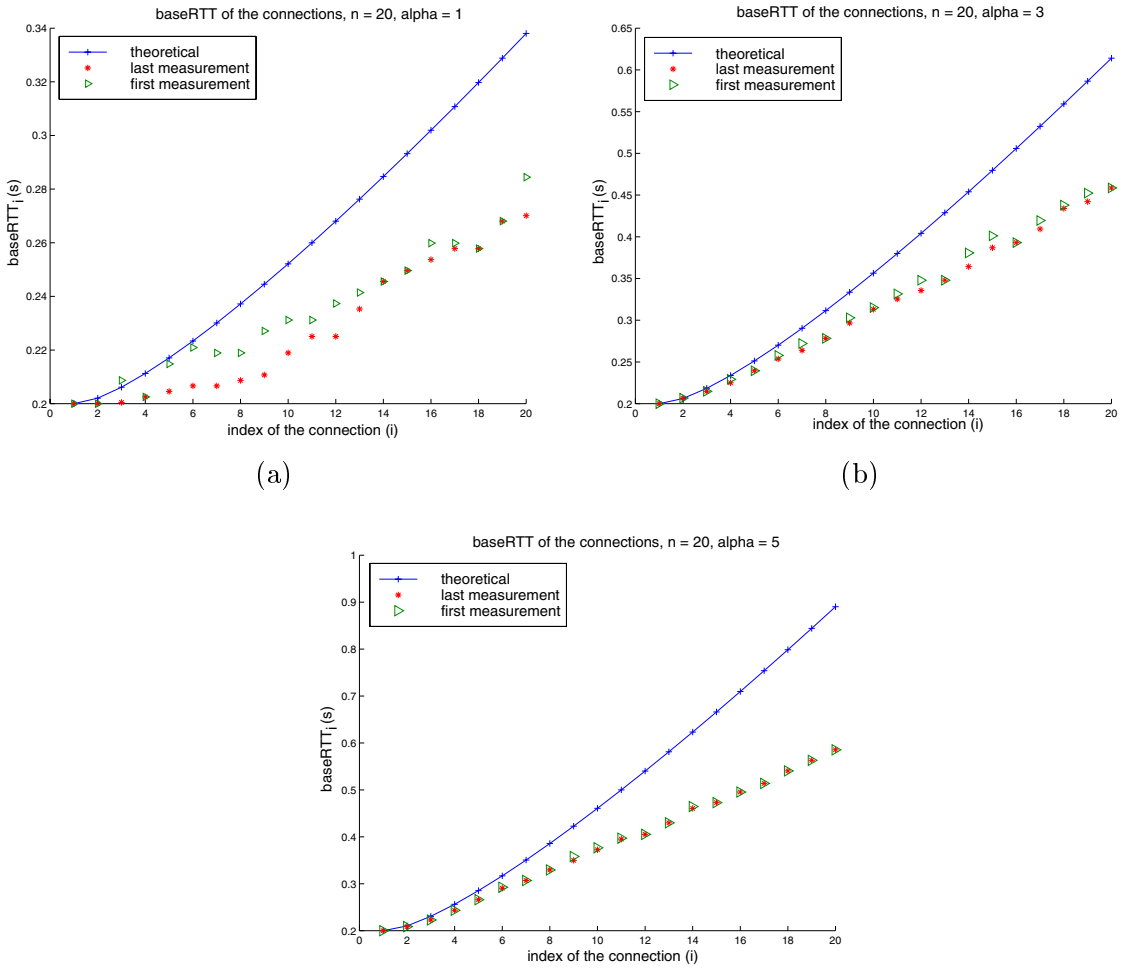

Figure 3: *baseRTT of the connections for (a)* $\alpha = 1$*, (b)* $\alpha = 3$ *and (c)* $\alpha = 5$.

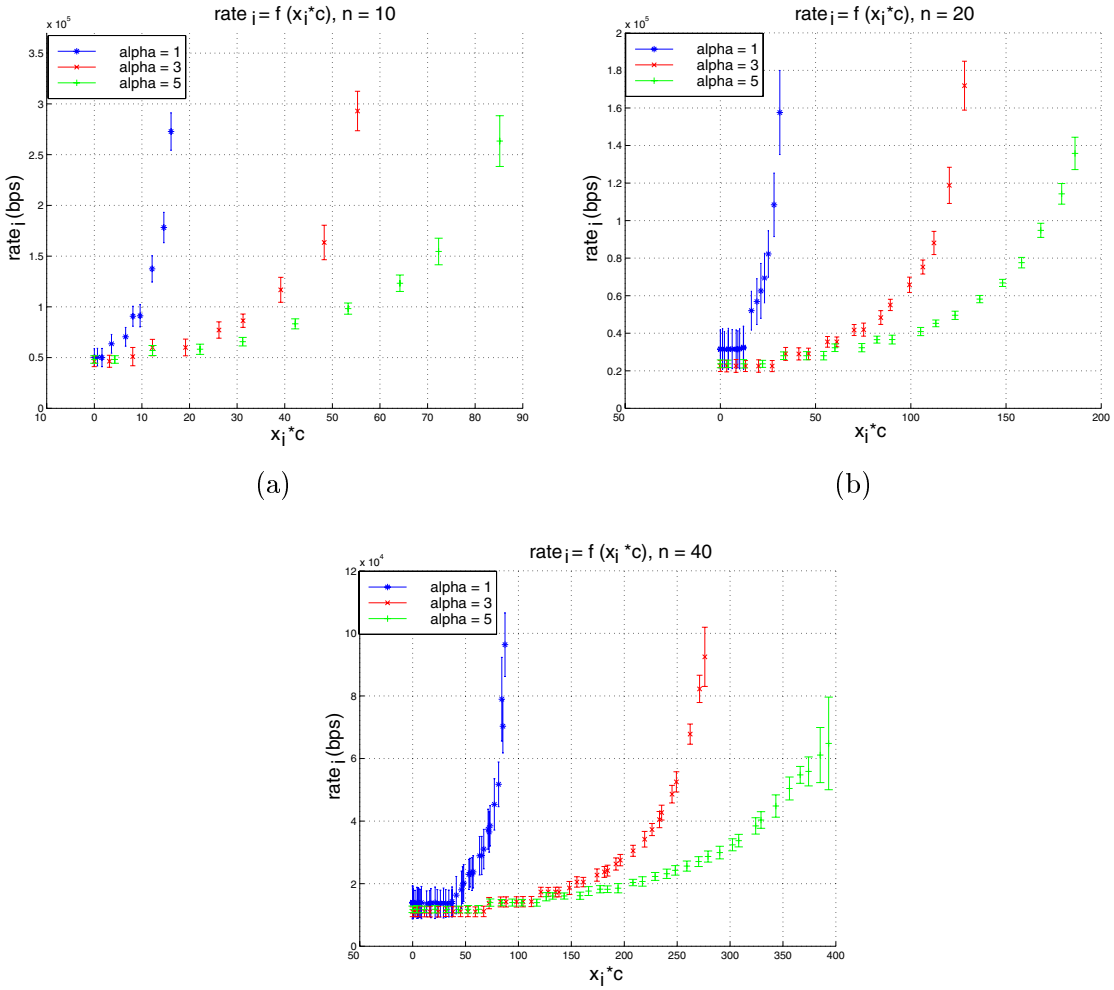

Figure 4: *Repartition of the rates for (a)* $n = 10$*, (b)* $n = 20$ *and (c)* $n = 40$*.*

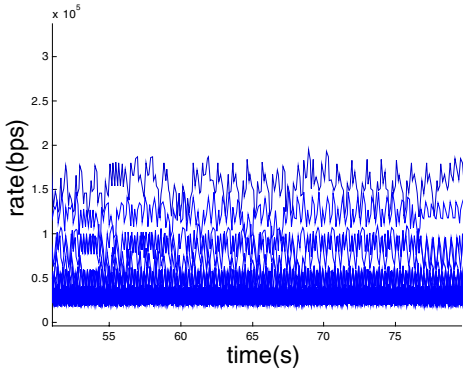

Figure 5: *Evolution of the rates over time for* $n = 20$ *and* $\alpha = 3$.

Another point of interest is the increase of the amplitude of the rate oscillations with the rate value. This is explained in the following. For all connections, the window size oscillates around its mean value, all oscillation amplitudes being similar. This leads to oscillations of the queuing delay Δ, which influences the rate value following:

$$\frac{\frac{\partial rate_i}{\partial \Delta}}{rate_i} = -\frac{1}{\Delta - x_i} \tag{9}$$

This derivative increases as x_i approaches Δ, explaining why the rate oscillations increase with the over-estimation factor.

The dynamics of the oscillations are illustrated in Figure 5 where the evolution of the rates of the connections as a function of the time is plotted (we chose a small simulation window to facilitate the reading of the plot). The oscillations' peaks are not synchronized in time and therefore don't lead to an under-utilization of the link capacity.

3.3 Conclusion for $\alpha = \beta$

We have shown that any over-estimation of the propagation delay of a connection results in an increase of its rate which gets worse as the over-estimation factor increases. We also have found that the rate oscillations did not allow to compensate this effect. As a result, the late connections, which have an important over-estimation factor, can get a lot more bandwidth than the earlier connections. Because of this, the enhanced TCP Vegas, when $\alpha = \beta$, does not achieve fairness among the connections. This leads to non deterministic transfer times.

4 Case 2: $\alpha < \beta$

We now turn to the case $\alpha < \beta$ in which stabilization of the window sizes, that would oscillate for $\alpha = \beta$ is observed. We now propose to analyze the joint impact of 1) the difference between α and β, and 2) the over-estimation of the propagation delay on the fairness of TCP Vegas. The network model and notations are the same as those depicted in section 3.

4.1 Analysis of the fairness

At the steady state, in the case $\alpha < \beta$, we can deduce from (1) that $\alpha \leq Diff_i \leq \beta$ for all i. Therefore, we can express the window sizes by:

$$\alpha \frac{d_i + \Delta}{\Delta - x_i} \leq cwnd_i \leq \beta \frac{d_i + \Delta}{\Delta - x_i} \tag{10}$$

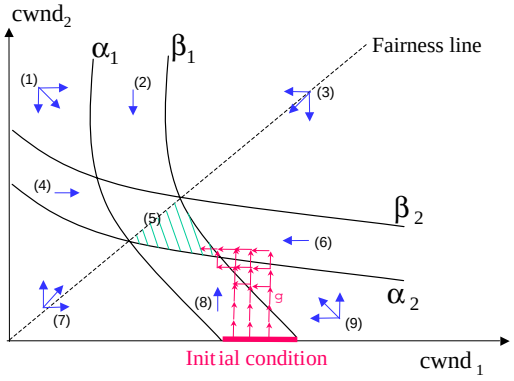

Figure 6: *Convergence region of TCP Vegas*

and derive the following expression for the throughput of connection i:

$$\frac{\alpha}{\Delta - x_i} \leq rate_i \leq \frac{\beta}{\Delta - x_i} \tag{11}$$

This equation holds the two reasons of unfairness of TCP Vegas. First, if the propagation delays are correctly estimated ($x_i = 0$), the rate of a connection converges to a value that lies between two bounds that depend on the parameters α and β. Therefore some connections could receive β/α times more bandwidth than other connections. Second, late connections will probably receive more bandwidth that earlier ones as the boundary values increase with over-estimation of the propagation delay. The adjective *probably* refers to the fact that the actual convergence value, because of possible overlap between boundaries of successive connections, can not be assessed to a greater value for later connections. Moreover, the convergence values depend on the arrival order of connections, as more than one solution exists to the equation $\sum_{i=1}^{n} rate_i = c$.

Let us now detail the case in which all connections measure accurately the propagation delay. We state that earlier connections will be favored and will receive more bandwidth, as shown in the heuristic argument that follows.

Let us consider the simple case where only two connections are sharing a bottleneck link. Figure 6 illustrates the convergence region of TCP Vegas for 2 users, but the same geometric picture can be easily extended to a case with more users. In the figure, α_i and β_i lines for connection i denote the sets of window size pairs $\{(cwnd_1, cwnd_2)|Diff_i = \alpha\}$ and $\{(cwnd_1, cwnd_2)|Diff_i = \beta\}$, respectively. The fairness line represents window size pairs of equal throughputs of connections, i.e. $\{(cwnd_1, cwnd_2)|rate_1 = rate_2\}$. Connection 1 increases its window size in regions (1), (4) and (7), and decreases it in regions (3), (6), and (9). Similarly, user 2 increases its window size in regions (7), (8) and (9), and decreases it in regions (1), (2) and (3). The only region where neither user updates its window size is region (5). The arrows in other regions indicate the directions in which the window sizes may get updated. Now, if we suppose that connection 1 starts first and connection 2 joins the network when connection 1 is in steady state, the initial conditions of the system are situated on the x-axis between the lines α_1 and β_1. And, starting from that region, the window sizes will converge to a point in the hachured part of region (5), assuming that the distance between α_i and β_i lines is sufficiently large compared to the amount ($\delta > 0$) by which users update their window sizes. In the hachured part of region (5), the rate of connection 1 is greater than the one of connection 2 and this explains the bias in favor of early connections. Of course, the greater the difference between α and β, the greater will be the unfairness.

In the next section, we present some simulation results that illustrate our analysis.

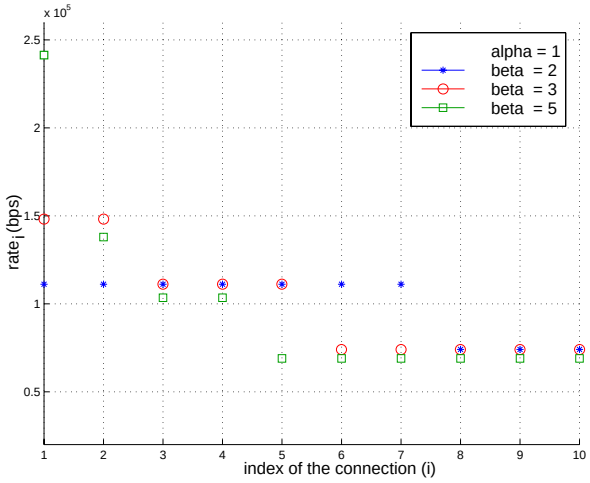

Figure 7: *Rate distribution between the connections for* $n = 10$ *and without propagation delay over-estimation*

4.2 Simulations

Using the setup of section 3.2.1, with $\alpha < \beta$, we simulated two scenarios: in the first one, we imposed that all connections have an accurate estimation the propagation delay ($x_i = 0$) while in the second one, more realistic, the propagation delay is estimated by the connections.

4.2.1 Case 1: without over-estimation of the propagation delay

Figure 7 illustrates the rate distribution between users for different values of the parameters (α, β). The x-axis and y-axis represent respectively the index and the rate of the connections. As expected, we see that early connections receive more bandwidth than later ones. Moreover, the unfairness increases with the ratio β/α. We can also notice that the oscillations disappeared.

4.2.2 Case 2: with over-estimation of the propagation delay

Now, we investigate the joint impact of α being unequal to β and the propagation delay over-estimation.

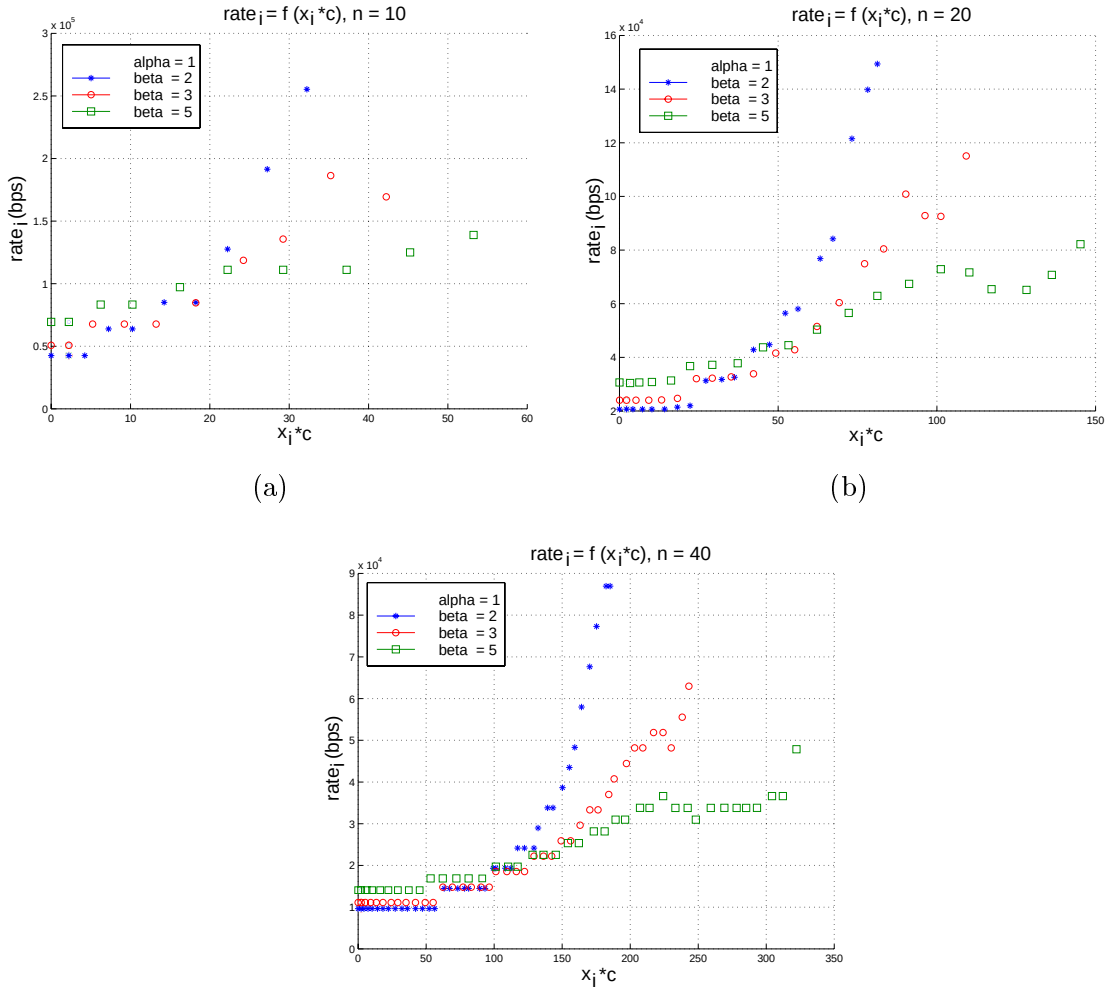

Figure 8: *Repartition of the rates for (a)* $n = 10$*, (b)* $n = 20$ *and (c)* $n = 40$.

In figure 8 (a,b,c), we plotted the rates of the connections as a function of their over-estimation factor, for different values of (α, β), and n. We see that the two effects tend to compensate each other and so the overall fairness increases as β furthers off α. However, the effects do not cancel out as the influence of the over-estimation factor dominates. This can be seen in the figure as the rates increase with increasing values of x_i.

4.3 Conclusion for $\alpha < \beta$

Under this setting, the rate of a connection converges to a stable value that depends on the arrival order of the connections. When the propagation delays are properly estimated, the earliest established connections are favored and receive more bandwidth. On the other hand, the later connections over-estimate the propagation delays and therefore gain a larger portion of the bandwidth. These two effects tend to counterbalance each other but the second tends to dominate.

5 Final Conclusion

In this article, we have studied the fairness of TCP Vegas. We have considered the two cases $\alpha = \beta$ and $\alpha < \beta$.

When $\alpha = \beta$, any over-estimation of the propagation delay of a given connection results in an increase of its rate that gets greater as the over-estimation factor increases. The rate oscillations do not allow for compensation of this effect. This results in unfair distribution of bandwidth among the users.

In the case $\alpha < \beta$, we showed that two reasons of unfairness of TCP Vegas are 1) the over-estimation of the propagation delay of a connection and 2) the fact that $\alpha \neq \beta$. The analysis of these two factors evidenced that, although their effects counterbalance, they do not cancel each other out. The over-estimation problem is dominant and causes unfairness.

Our final conclusion is that the use of TCP Vegas in the future (instead of Reno) should rely on $\alpha = \beta$ but will require that propagation delays be correctly estimated. There is no obvious way to achieve this.

Acknowledgement

The authors would like to thank Professor Patrick Thiran for fruitfull discussions.

References

[1] J. S. Ahn, P. B Dansig, Z. Liu, , and L. Yan. Evaluation of TCP Vegas: Emulation and experiment. In *ACM SIGCOMM Computer Communication Review*, volume 25, pages 185–195, 1995.

[2] Thomas Bonald. Comparison of TCP Reno and TCP Vegas via fluid approximation. Technical report, INRIA, 1998.

[3] L.S. Brakmo and L.L. Peterson. TCP Vegas: End to end congestion avoidance on a global internet. *IEEE J. of Selected Areas in Communication*, 13:1465–1480, 1995.

[4] G. Hasegawa, M. Murata, and H. Miyahara. Fairness and stability of congestion control mechanisms of TCP. In *Globecom'99*, pages 1329–1336.

[5] G. Hasegawa, M. Murata, and H. Miyahara. Fairness and stability of congestion control mechanisms of TCP. In *11th ITC Specialist Seminar*, pages 255–262, October 1998.

[6] J. Mo, R.J. La, V. Anantharam, and J. Walrand. Analysis and comparison of TCP Reno and Vegas. In *Globecom'99*.