Inter-protocol fairness between TCP New Reno and TCP Westwood+

Niels Möller[*], Chadi Barakat[†], Konstantin Avrachenkov[†], and Eitan Altman[†]
[*]KTH, School of Electrical Engineering
SE-100 44, Sweden
Email: niels@ee.kth.se
[†]INRIA
2004 route des Lucioles, 06902 Sophia Antipolis, France
Email: {Chadi.Barakat|k.avrachenkov|Eitan.Altman}@sophia.inria.fr

***Abstract*— In this paper we investigate the effect of introducing TCP Westwood+ on regular TCP New Reno. By means of analytical modeling and ns-2 simulations, we demonstrate that the two protocols get different shares of the available bandwidth in the network. Our main result is that the bandwidth sharing between the two protocols depends on one crucial parameter: the ratio between the bottleneck router buffer size and the bandwidth delay product. If the ratio is smaller than one, TCP Westwood+ takes more bandwidth. On the contrary, if the ratio is greater than one, it is TCP New Reno which gets the larger part. Inspired by our results, we propose a simple modification to the window decrease algorithm in TCP Westwood+ that solves the unfairness problem for large buffer sizes. For small buffers, the unfairness problem is still open.**

I. Introduction

In recent years, several new proposals and implementations of TCP congestion control algorithms have been developed, motivated by a growing heterogeneity of networks such as wireless networks, high speed networks, ad-hoc and sensor networks. In all these types of networks, the transmission is subject to losses due to unreliable links (transmission losses) in addition to losses due to congestion (congestion losses). The Westwood+ TCP version [1] has appeared to be particularly useful when transmission losses cannot be neglected. TCP Westwood+ is novel with respect to TCP Westwood because of a new, simpler and unbiased estimator of the available bandwidth. It behaves exactly as TCP New Reno version in increasing its window when there are no packet losses. However when a loss occurs, the behavior is different: instead of employing the classic TCP by half window decrease, Westwood+ decreases the window size to a new value that exactly matches the bandwidth available at the time of congestion. In particular, the window size is set equal to the available bandwidth times the smallest RTT it has been observed so far. The rationale for this choice is to keep full the "available pipe", where the available pipe is the available bandwidth times the minimum round trip time.

When evaluating proposed TCP improvements, one important issue is that of fairness, and in particular fairness in a mixed environment where the old and new TCP versions coexist and share the same resources. The major goal of the present work is to study the inter-protocol fairness between TCP Westwood+ and TCP New Reno. As TCP Westwood+ can reduce the congestion window less than TCP New Reno, one might suspect that TCP Westwood+ takes a larger share of the available resources. On the other hand, it is known that TCP New Reno under-utilizes the bandwidth in the case of small buffer sizes. So does TCP Westwood+ just take this un-used part of the bandwidth?

In [2], the impact of the buffer size on the performance of TCP Westwood+ was studied, and it was shown that unlike TCP New Reno, Westwood+ can achieve full link utilization with arbitrarily small link buffers. Thus, one can expect that the buffer size at the bottleneck will be one of the most important parameters in the analysis. Moreover, the problem of the choice of the buffer size for regular TCP traffic has drawn a significant attention [3]–[7]. Here we show that the router buffer size has a significant influence not only on the efficiency of the network but also on the fairness between different TCP versions. In particular, we show that in order TCP Westwood+ to take a fair share of the available bandwidth one needs to choose the buffer size of the bottleneck router not smaller than half of the bandwidth-delay product.

We provide a model that describes the interaction of TCP Westwood+ with the bottleneck router buffer as well as with the conventional TCP New Reno. Since queueing delays and packet losses due to congestion are coupled to the queue and window sizes, it becomes necessary to take queue and buffer size into account. The evolution of the window sizes of both flows is modeled as a hybrid system [8], with a continuous increase between congestion events, and discontinuous reductions at the congestion event. This model lets us calculate analytically the throughput for the flows for any buffer size. The results are also validated using `ns2` simulations.

Both the analytical model and simulations give the same results: If the buffer size equals the bandwidth-delay product, TCP Westwood+ and TCP New Reno will share available capacity almost equally. For smaller buffers, Westwood+ wins the battle for capacity, and for larger buffers, TCP New Reno wins. Furthermore, we show that the results are only sensitive to a single parameter: the ratio between the buffer size and

The work was supported by the France Telecom R&D Grant "Modélisation et Gestion du Trafic Réseaux Internet" no. 42937433.

C	Capacity of bottleneck link	[bytes/s]
B	Buffer size	[bytes]
T_i	Round trip time of flow i, excluding queueing delay at the bottleneck	[s]
m_i	Segment size of flow i	[bytes]
$q(t)$	Queue size	[bytes]
$w_i(t)$	Window size of flow i	[bytes]
$r_i(t)$	Sending rate of flow i	[bytes/s]

TABLE I
NOTATION

the bandwidth delay product. Finally, we propose a simple modification to TCP Westwood+ that makes TCP Westwood+ efficient even in the case of large buffers.

The paper is organized as follows. In the next Section II we describe the system model and in the ensuing Section III we provide its mathematical analysis. Then, in Section IV we provide the numerical as well as simulation results and discussion. We conclude the paper with Section V.

II. SYSTEM MODEL

We use a hybrid fluid flow model for a network with two competing TCP sources: TCP New Reno and TCP Westwood+. They share a single bottleneck router with a drop-tail buffer of size B and with transmission capacity C. The state variables of interest are the queue size $q(t)$ at the bottleneck router and the congestion window sizes $w_i(t), i = 1, 2,$ of each TCP source. We assign index 1 to the TCP New Reno flow and index 2 to the TCP Westwood+ flow. The variable $q(t)$ is regarded as continuous and $w_i(t)$ as piece-wise continuous, and the sending rate of each sources is one full window of data per round trip time, where the round trip time consists of a constant propagation delay T_i and a queueing delay $q(t)/C$ that is determined by the current queue size. The window size corresponds to the amount of data that a source has transmitted, but not yet received any acknowledgment (ACK) for. The notations are summarized in Table I.

For the window dynamics, we focus on the congestion avoidance mode of TCP. In TCP New Reno, congestion avoidance is based on Additive increase / Multiplicative decrease: The window is increased by one packet per round trip time as long as ACKs are received. When a packet loss is detected, the window size is set to one half of the value before the loss.

In TCP Westwood+ [1], the additive increase is the same. The difference is that the multiplicative decrease is replaced by a different mechanism based on estimation of the available bandwidth and of the propagation delay. The original motivation was to improve TCP performance over wireless links, with a significant rate of losses that are due to transmission errors rather than congestion. When Westwood+ detects a packet loss, it sets its window size to the product of the estimated bandwidth before the loss, and the end-to-end propagation delay. The idea is that this window size is sufficiently small to allow queues on the path to drain, but not smaller.

The propagation delay estimate is simple: it is just the minimum observed round trip time. As long as at least one packet has been sent when the queue is close to empty, this estimate is accurate. The bandwidth estimation is more complex. We have to refer to the Westwood literature for the details [1], but the main idea is to obtain "bandwidth samples" from the stream of ACKs, and form the estimate by low-pass filtering of these samples. The first proposed version of Westwood formed a bandwidth sample for each ACK, while Westwood+ collects a round trip time worth of ACKs before forming a bandwidth sample.

Thus, according to the fluid flow approximation of window based congestion control schemes such as TCP New Reno and TCP Westwood+, the sending rate of a flow is one full window per RTT:

$$r_i(t) = \frac{w_i(t)}{T_i + q(t)/C}. \tag{1}$$

The queue size is related to the sending rates by $\mathrm{d}q(t)/\mathrm{d}t = \sum r_i(t) - C$, or, substituting (1),

$$\frac{\mathrm{d}q(t)}{\mathrm{d}t} = \sum_i \frac{w_i(t)}{T_i + q(t)/C} - C. \tag{2}$$

The above two equations ignore signaling delays, but it includes the dependence on the queue size, via the RTT $T_i + q(t)/C$. Since the queue size must be non-negative, (2) is valid only when $q(t) > 0$ or the right hand side is positive; otherwise $\mathrm{d}q/\mathrm{d}t = 0$.

The evolution of the window sizes is governed by the additive increase of TCP's congestion avoidance mode, which is the same for both New Reno and Westwood+. The window is increased by one packet, m_i bytes, each RTT.

$$\frac{\mathrm{d}w_i(t)}{\mathrm{d}t} = \frac{m_i}{T_i + q(t)/C}. \tag{3}$$

At the congestion event, the flow or flows that lose packets make a discontinuous change in its window size. For TCP New Reno, $w_1(t^* + 0) = w_1(t^* - 0)/2$. For TCP Westwood+, $w_2(t^* + 0) = \mathrm{RTT}_{\min}\hat{c}_2$, where $\hat{c}_2$ is the estimate of the flow's bandwidth, and $\mathrm{RTT}_{\min}$ is the smallest observed RTT.

For TCP Westwood+, there are several issues with how this should be modeled.

- $\mathrm{RTT}_{\min}$: This is intended to be the round trip delay, excluding queueing delay. So it makes sense to put $\mathrm{RTT}_{\min} = T$. But on the other hand, if the Westwood+ flow is started when the bottleneck is already loaded, Westwood cannot observe T, and it may be more accurate to set $\mathrm{RTT}_{\min} = T + \min q(t)/C$.
- $\hat{c}_2$: The bandwidth is "sampled" once per round trip time, and then these samples are low-pass filtered to form a smoother estimate. The simplest model is to put $\hat{c}_2 = r_2(t^* - 0) = w_2(t^* - 0)/(T + B/C)$. This is an assumption of optimistic estimation, and it neglects the delay and the bias which are present in the real filter.

With $\mathrm{RTT}_{\min} = T$ and $\hat{c}_2 = w_2(t^* - 0)/(T + B/C)$, it follows that $w_2(t^* + 0) = \beta w_2(t^* - 0)$, with the constant $\beta = CT/(CT + B)$.

III. Analysis

To analyze the system evolution, we use separate models for the evolution between congestion events, and for the congestion events. A *congestion event* is a short period of time when the router queue is full, and one or more packets are dropped. Finally, we put these two models together to find the stationary behavior, and the corresponding throughput.

A. System evolution between congestion events

Assume that a congestion event ends at time $t = 0$. At this time $q(0) = B$, i.e., the buffer is full, and the window sizes are given by the initial conditions $w_i(0) = w_i^0$.

After a congestion event, the evolution of the rates and the queue can be divided into three phases.

- Phase 1: During the first phase, the total rate, $r_1 + r_2$, is smaller than C, and increasing. In this phase, the queue is shrinking, and it may even become empty.
- Phase 2: During the second phase (which is present only for small buffer sizes), the queue is empty, and sending rates are increasing. During the second phase, the link is under-utilized, and the phase ends when the total sending rate reaches the link capacity, $r_1 + r_2 = C$.
- Phase 3: During the third phase, the ACK-clock mechanism forces the total sending rate to stay essentially constant, barely larger than C, and the growing windows result in a growing queue, not increasing sending rates. The third phase is terminated by the next congestion event, which happens when the queue size reaches the buffer size, $q(t) = B$.

The objective of the analysis in this section is to find the smallest $t^* > 0$ such that $q(t^*) = B$, and to express t^* and the corresponding window sizes $w_i(t^*)$ as functions of the initial window sizes w_i^0. These functions are needed in Sec. III-B, when deriving the evolution over a large number of congestion events.

When investigating fairness between two flows, using $T_1 \neq T_2$ would introduce a prejudice, favoring one of the flows against the other. To give the two flow equal opportunities, we assume $T_1 = T_2 = T$. This assumption lets us to introduce a virtual time, which can be thought of as measuring time in number of round trips. This tool makes it possible to find explicit solutions to the differential equations.

Virtual time s is defined by $\mathrm{d}t = (T + q(t)/C)\mathrm{d}s$. This change of variables transforms Equations (2) and (3) into

$$\frac{\mathrm{d}w_i(s)}{\mathrm{d}s} = m_i \tag{4}$$

$$\frac{\mathrm{d}q(s)}{\mathrm{d}s} = -q(s) - CT + \sum_i w_i(s) \tag{5}$$

The second equation is still valid only when $q(s) > 0$ or the right hand side is positive.

Let s^* denote the virtual time corresponding to t^*, i.e., the next congestion event. Given s^*, the amount of data that is transmitted up to time t^* can be computed as follows. First solve (4), which gives,

$$w_i(s) = w_i^0 + sm_i \tag{6}$$

Then the volume of data transmitted up to time s^* is

$$D_i = \int_0^{t^*} r(t)\mathrm{d}t = \int_0^{s^*} w(s)\mathrm{d}s = s^*\left(w_i^0 + \frac{m_i s^*}{2}\right) \tag{7}$$

This says simply that the amount of data is the number of round trips, s^*, times the average window size. Here, the "average" is not a proper time average, but an average with respect to the virtual time s. We will compute the throughput as D_i/t^*. The calculation of s^* and t^* depends on the buffer size.

In general, t^* can computed given $q(s)$ and s^*, by integrating

$$t^* = \int_0^{t^*} \mathrm{d}t = \int_0^{s^*} \frac{\mathrm{d}t}{\mathrm{d}s}\mathrm{d}s = Ts^* + \frac{1}{C}\int_0^{s^*} q(s)\mathrm{d}s \tag{8}$$

The calculations in absolute time, t, are simplified by the remarkable fact that (3) and (2) can be solved for q in terms of w. Substitution of (3) into (2) gives

$$\begin{aligned}\frac{\mathrm{d}q(t)}{\mathrm{d}t} &= -C + \sum_i \frac{w_i(t)}{m_i}\frac{\mathrm{d}w_i(t)}{\mathrm{d}t} \\ &= \frac{\mathrm{d}}{\mathrm{d}t}\left\{-Ct + \sum_i \frac{1}{2m_i}(w_i(t))^2\right\}\end{aligned} \tag{9}$$

Integrating, we get

$$q(t_2) - q(t_1) = -C(t_2 - t_1) + \frac{1}{2}\sum_i \frac{(w_i(t_2))^2 - (w_i(t_1))^2}{m_i} \tag{10}$$

For any time interval, during which the queue stays non-empty, this allows us to compute the length of the interval given only the initial and final state,

$$t_2 - t_1 = \frac{1}{C}\left\{q(t_1) - q(t_2) + \frac{1}{2}\sum_i \frac{(w_i(t_2))^2 - (w_i(t_1))^2}{m_i}\right\} \tag{11}$$

1) Full utilization: The link will be fully utilized if and only if $q(t) > 0$ for all time (except possible for an isolated instant). In other words, the second phase is empty. This section derives conditions on the initial conditions for this to happen.

Introduce the notation $W = w_1^0 + w_2^0$ and $M = m_1 + m_2$. Substituting $w_i(s) = w_i^0 + sm_i$ into (5) gives

$$\frac{\mathrm{d}q(s)}{\mathrm{d}s} = -CT - q(s) + W + sM \tag{12}$$

which together with the initial condition $q(0) = B$ can be solved explicitly,

$$q(s) = Be^{-s} + (W - CT)(1 - e^{-s}) + (e^{-s} - 1 + s)M \tag{13}$$

This equation is valid only as long as $q > 0$, since the differential equation doesn't model queue underflow. Looking more closely at this equation, it can be divided into a transient, related to the initial buffer B and a new "equilibrium size" $W - CT$, and a linear growth with rate M. Asymptotically,

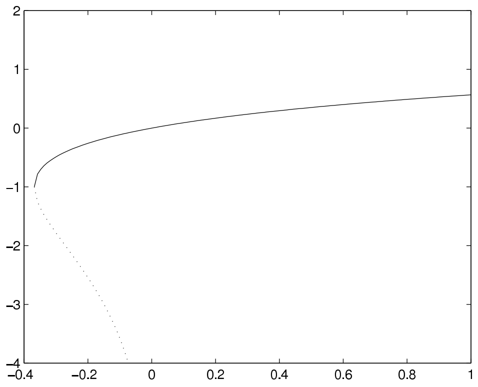

Fig. 1. The two real branches of Lambert's W function. W_0 (solid) is defined for all $x \geq -1/e$, while W_{-1} (dotted) is defined on the interval $-1/e \leq x < 0$.

for large s, we have $q(s) \approx W + (s-1)M - CT$, i.e., the queue is the difference between the total window size one RTT earlier, and the bandwidth-delay product CT.

Proposition 1: The link is fully utilized if and only if either of these two inequalities is satisfied:

$$W \geq CT \tag{14}$$

$$\frac{B}{M} \geq \exp\left(\frac{CT-W}{M}\right) - 1 - \frac{CT-W}{M} \tag{15}$$

Proof: First observe that if $W \geq CT$, then $q(s) > 0$ for all s (note that $e^{-s} - 1 + s > 0$). So assume that $W < CT$. Then the function $q(s)$, defined by (13), is initially decreasing, and has a single minimum at $s_0 = \log(1 + (CT + B - W)/M)$. We have full utilization if and only this minimum value $q(s_0)$ is non-negative. By substituting s_0 into (13), we see that $q(s_0) \geq 0$ is equivalent (15), which concludes the proof. ■

2) Large buffer: If the buffer is "large", i.e., the condition in Prop. 1 is satisfied, then the queue never underflows, and we can find s^* by putting $q(s) = B$ in (13). The solution can be expressed in terms of Lambert's function [9], defined as the inverse of $z \mapsto ze^z$. We use the two real branches, denoted W_0 and W_{-1}, and illustrated in Fig. 1.

Proposition 2: When the full-utilization condition of Prop. 1 is satisfied, the values of s^* and t^* are given by

$$s^* = \tilde{s} + 1 + W_0\left(-(1+\tilde{s})e^{-(1+\tilde{s})}\right) \tag{16}$$

$$t^* = \frac{s^*}{C}\left(W + \frac{Ms^*}{2}\right) \tag{17}$$

where $\tilde{s}$ is defined by

$$\tilde{s} = (CT + B - W)/M \tag{18}$$

Proof: Put $q(s) = B$ in (13). After some simplifications, this equality implies

$$1 + \tilde{s} = (1+\tilde{s})e^{-s} + s$$

This equation has two solutions, the trivial one $s = 0$, and a second solution which can be expressed using the W_0 function, resulting in (16).

With this value for s^*, (17) follows from (11). ■

The throughput can be computed from (7),

$$\frac{D_i}{t^*} = C\ \frac{w_i^0 + m_i s^*/2}{W + Ms^*/2} \tag{19}$$

As expected, with full utilization, the total throughput, $(D_1 + D_2)/t^*$, equals C.

3) Small buffers: When computing the throughput for a small buffer, we must handle the three phases separately, since neither the differential equation for q, nor the time interval equation (11), is valid during the second phase, when the queue is empty and the link is under-utilized.

Proposition 3: When the full-utilization condition of Prop. 1 is *not* satisfied, the values of s^* and t^* are given by

$$s^* = \tilde{s} + 1 + W_0\left\{-e^{-1-B/M}\right\} \tag{20}$$

$$t^* = \frac{Ws_1 + CT(s_3 - s_1) + M(s_1^2 + s_3^2)/2}{C} \tag{21}$$

where s_1 and s_2 are defined by

$$s_1 = 1 + \frac{CT-W}{M} + W_{-1}\left(-(1+\tilde{s})e^{-1-(CT-W)/M}\right)$$

$$s_3 = 1 + \frac{B}{M} + W_0\left(-e^{-1-B/M}\right)$$

Proof: Denote the duration of the three phases, in absolute time and virtual time, by t_1, t_2, t_3, s_1, s_2, and s_3. We handle one phase at a time.

The first phase: We find s_1 by putting $q(s_1) = 0$ in (13) and solving for s_1. Then t_1 is found by substituting initial and final state in (11). This procedure gives:

$$s_1 = 1 + \frac{CT-W}{M} + W_{-1}\left(-(1+\tilde{s})e^{-1-(CT-W)/M}\right)$$

$$t_1 = \frac{B + Ws_1 + Ms_1^2/2}{C}$$

The second phase: Throughout this phase, $q(0) = 0$. The phase starts with $\sum w_i(t) < CT$, and ends when $\sum w_i(t) = CT$. Initially, the total window size is $W + s_1 M$, so it follows that

$$s_2 = \frac{CT-W}{M} - s_1$$

$$t_2 = Ts_2$$

The third phase: This is similar to the first phase, but with different start and stop conditions. Initially, $\sum w_i(s_1 + s_2) = CT$ and $q(s_1 + s_2) = 0$. With these initial condition, the differential equations (4) and (5) have the solution

$$q(s) = \left(e^{-(s-s_1-s_2)} + (s - s_1 - s_2) - 1\right)M \tag{22}$$

The phase ends when $q(s) = B$. The solution of this equation, which is found using the same method as for the first phase, gives the value of s_3. t_3 is found by substituting initial and final state in (11). The result is

$$s_3 = 1 + \frac{B}{M} + W_0\left(-e^{-1-B/M}\right)$$

$$t_3 = \frac{-B + CTs_3 + Ms_3^2/2}{C}$$

Finally, addition of the values for each phase, $s^* = s_1+s_2+s_3$ and $t^* = t_1 + t_2 + t_3$, results in (20) and (21). ■

4) Throughput: Propositions 1–3 let as compute t^* and s^* as functions of the initial window sizes. With these values, the data volume D_i follows from (7), and the throughput D_i/t^* for each flow can be computed. For reference in the next section, we define the functions $\tilde{s}(W)$ and $s^*(W)$ as follows:

$$\tilde{s}(W) = \frac{CT + B - W}{M} \tag{23}$$

$$s^*(W) = \tilde{s} + 1 + \begin{cases} W_0\left(-(1+\tilde{s})e^{-(1+\tilde{s})}\right) & \text{full utilization} \\ W_0\left(-e^{-1-B/M}\right) & \text{otherwise} \end{cases} \tag{24}$$

B. System evolution at congestion events

The previous section analyzed the queue evolution between congestion events. To find the average throughput over a longer time, we need to know the stationary behavior that results from a large number of congestion events. We first consider what happens at the single congestion event at time $t = t^*$.

As shown at the end of Section II, at the congestion event if TCP New Reno loses a packet: $w_1(t^*+0) = 0.5w_1(t^*-0)$, and if TCP Westwood+ source loses a packet: $w_2(t^* + 0) = \beta w_2(t^* - 0)$, with the constant $\beta = CT/(CT + B)$.

Then, an important issue is *which* flow loses packets at a congestion event. It could be one of the flows, or both. In the TCP fairness literature, it is common to use stochastic modeling for this (see e.g., [10], [11]).

Under the fairly general assumption that the probabilities of the different possible outcomes at a congestion event depends only on the window sizes of the involved flows just prior to the congestion, the evolution can be described as a Markov chain. Let X^k denote the vector of the two window sizes just *after* a congestion event. If the first flow uses TCP New Reno and the second uses TCP Westwood+, the evolution can be described as

$$X^{k+1} = \mathcal{D}_{\ell_k} G(X^k) \tag{25}$$

In this equation

$$G\begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} + s^*(w_1 + w_2)\begin{pmatrix} m_1 \\ m_2 \end{pmatrix} \tag{26}$$

represents the window growth between congestion events, and the function $s^*(W)$ is defined by (24). The actual packet loss is represented by the random variable ℓ_k, with three possible values 0, 1, and 2. The window reduction is represented by the constant diagonal matrices $\mathcal{D}_i$,

$$\mathcal{D}_0 = \begin{pmatrix} 1/2 & 0 \\ 0 & \beta \end{pmatrix} \quad \mathcal{D}_1 = \begin{pmatrix} 1/2 & 0 \\ 0 & 1 \end{pmatrix} \quad \mathcal{D}_2 = \begin{pmatrix} 1 & 0 \\ 0 & \beta \end{pmatrix} \tag{27}$$

Of the three outcomes, $\mathcal{D}_0$ represents a loss event where each flow loses a packet, $\mathcal{D}_1$ represents an event where only the first flow loses a packet, and $\mathcal{D}_2$ represents an event where only the second flow loses a packet.

If we make the further assumption that the probability that $\ell_k = j$ depends only on the state X^k, i.e.,

$$\mathrm{P}(\ell_k = j | X^k, \ell_{k-1}, \ell_{k-2}, \ldots) = p_j(X^k) \tag{28}$$

then (25) clearly describes a Markov chain.

To gain insight into the problem, and to be able to solve it analytically, we have to make a few simplifying assumptions and approximations. Our first assumption is that flows are fully synchronized, i.e., that at every congestion event, both flows lose packets. This means that $\ell_k = 0$ for all k, and it actually makes the process fully deterministic.

The second simplification is dropping the non-linear terms of the function $s^*(w_1, w_2)$. In both expressions for s^*, (16) and (20) are of the form $s^* = \tilde{s} + 1 + W_0(\cdots)$, where the final term is non-linear, and bounded between -1 and 0. So by replacing s^* by $\tilde{s} + 1/2$, the error in $G(X^k)$ is at most half a packet.

To give both flows equal opportunities, we also put $m_1 = m_2 = m$. The result is the following approximation of the evolution of the state X^k.

$$\begin{aligned} X^{k+1} &= \mathcal{D}_0\left(X^k + (\tilde{s}(X_1^k, X_2^k) + 1/2)m\begin{pmatrix} 1 \\ 1 \end{pmatrix}\right) \\ &= \begin{pmatrix} 1/4 & -1/4 \\ -\beta/2 & \beta/2 \end{pmatrix} X^k + (CT + B + m)\begin{pmatrix} 1/4 \\ \beta/2 \end{pmatrix} \end{aligned} \tag{29}$$

Since $\beta < 1$, the matrix has all eigenvalues within the unit circle. Then, for any initial values, the X^k sequence converges to

$$\begin{aligned} X^* &= \left(I - \begin{pmatrix} 1/4 & -1/4 \\ -\beta/2 & \beta/2 \end{pmatrix}\right)^{-1} (CT + B + 1)\begin{pmatrix} 1/4 \\ \beta/2 \end{pmatrix} \\ &= \frac{CT + B + m}{3 - 2\beta}\begin{pmatrix} 1 - \beta \\ \beta \end{pmatrix} \end{aligned} \tag{30}$$

To find the throughput for both flows in the stationary regime, we plug in the window sizes X^* as the initial window sizes in the procedure of Sec. III-A.4.

C. Limits

Let us study the fairness in the limiting cases when buffers are very small or very large. Of particular interest is the throughput ratio

$$\frac{D_2}{D_1} = \frac{2w_2^0 + ms^*}{2w_1^0 + ms^*} \tag{31}$$

Proposition 4: In the limit as $B \to \infty$, Westwood+ gets one quarter of the capacity, while New Reno gets three quarters. In the limit as $B \to 0$, Westwood+ gets all the capacity, and New Reno gets 0.

Proof: For the large buffer case, note that $CT + B = CT/\beta$. We parameterize the expressions in terms of β, and let $\beta \to 0$. We have from (30) that

$$\begin{pmatrix} w_1^0 \\ w_2^0 \end{pmatrix} = X^* = \frac{1}{3}\begin{pmatrix} CT/\beta + O(1) \\ CT + O(\beta) \end{pmatrix} \tag{32}$$

and $W = CT/(3\beta) + O(1)$. It is clear that we have full utilization, and we get

$$s^* = \frac{2CT}{3M\beta} + O(1) \tag{33}$$

Finally, $D_2/D_1 = (1 + O(\beta))/(3 + O(\beta)) \to 1/3$.

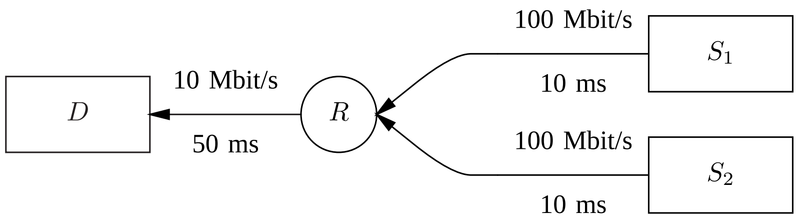

Fig. 2. Experimental setup. Parameter values correspond to scenario #1.

Scenario #	C [Mbit/s]	T [ms]	CT [packets]
0	1	50	4.2
1	10	50	42
2	1	200	16.7
3	10	200	167

TABLE II
NETWORK PARAMETERS FOR FOUR DIFFERENT SCENARIOS

For the small buffer case, first note that (30) implies that $W - CT \to m > 0$ as $B \to 0$, so by Prop. 1, we have full utilization for all sufficiently small B. Taking limits in Prop 2, we find that $s^* \to 0$ as $B \to 0$. Furthermore, Eq. (30) also implies that $w_1^0 \to 0$ and $w_2^0 \to CT + m$ as $B \to 0$ and $\beta \to 1$. It follows that $D_2/D_1 \to \infty$ as $B \to 0$. ■

This result is different from what one could obtain by using the throughput formulas for TCP New Reno and Westwood+ in [12]:

$$T^{\text{Reno}} = \frac{1}{\text{RTT}}\sqrt{\frac{2(1-p)}{p}}, \quad T^{\text{Westwood}} = \sqrt{\frac{1-p}{pT_q\text{RTT}}}. \qquad (34)$$

In these equations, T_q is the average queueing time and $\text{RTT} = T + T_q$ is the average round-trip time. Let's form the throughput ratio:

$$\frac{T^{\text{Westwood}}}{T^{\text{Reno}}} = \frac{1}{\sqrt{2}}\sqrt{1 + \frac{T}{T_q}} \qquad (35)$$

We see that this ratio predicts fairness when $T_q = T$, i.e., when the *average* queuing delay equals the propagation delay. This is slightly different from our model, which predicts fairness when the *maximum* queuing delay, B/C, equals the propagation delay.

In the small buffer limit, $T_q \to 0$, the ratio tends to infinity. So here we have perfect agreement. In the large buffer limit, $T_q \to \infty$, the ratio tends to $1/\sqrt{2}$, significantly higher than the prediction $1/3$ from our analysis. The simulation results presented in the next section will support our analysis in predicting the trend of the throughput ratio for large buffers. Our explanation for this discrepancy is in the random loss assumption made by the square root formula, which does not hold in our setting.

IV. Results and Discussion

We use the network topology illustrated in Fig. 2. There are two source nodes, S_1 (New Reno) and S_2 (Westwood+), sending data to a single destination node, D. There is one intermediate router, R, and the link between the router and the destination is the network's bottleneck. For the bottleneck link, we use two values for the capacity C, 1 Mbit/s and 10 Mbit/s, and two different values of the propagation delay, corresponding to round trip propagation delay T of 50 and 200 ms. Table II shows the parameter values for the four scenarios, and the corresponding bandwidth-delay product. Both sources use a packet size of 1500 bytes. For all scenarios, we vary the buffer size of the router between 2 packets and roughly twice the bandwidth-delay product, $2CT$.

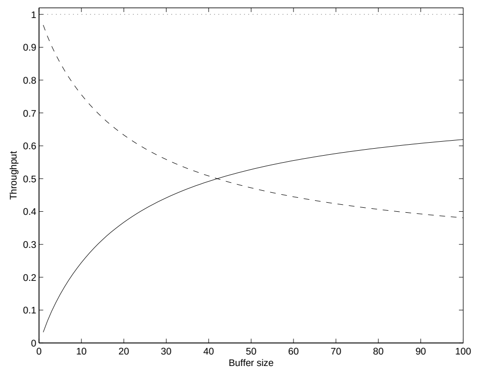

Fig. 3. Normalized throughput vs. buffer size (in packets) for scenario #1.

The resulting throughputs according to the analysis of Sec. III, are shown in Figs. 3 and 4.

In Fig. 3, we plot the normalized throughput (a fraction of the capacity of the bottleneck link) for scenario #1. The solid line represents TCP New Reno, and the dashed line represents TCP Westwood+. The topmost dotted curve is the sum of the normalized throughputs, i.e, the link utilization. In [2] it was shown that Westwood+ achieves almost full utilization for arbitrary small buffer sizes, so it is not surprising that TCP Westwood+ dominates over New Reno for small buffer sizes. When the buffer size equals the bandwidth-delay product, 42 packets in scenario #1, both flows get the same throughput. This is expected from the model, since for this buffer size, $\beta = 1/2$, both flows use the same decrease factor after a loss, and both elements of X^* are equal.

For larger buffer sizes, Westwood+ suffers from its estimation of $\text{RTT}_{\min}$. It uses the smallest observed RTT, 50 ms, even though in stationarity, the buffer never gets empty, and the actual RTT stays significantly higher than Westwood's $\text{RTT}_{\min}$ at all times. This forces Westwood+ to reduce its window size, after a packet loss, even more than New Reno does.

In Fig. 4, the corresponding curves for all four scenarios are shown in the same figure. To aid the comparison, the horizontal axis has been scaled so that for each scenario, 1 corresponds to the bandwidth-delay product. The theoretical throughput curves for all four scenarios are plotted on top of each other, the increasing curve represents New Reno, the decreasing curve represents Westwood+, and the uppermost curve close to one is the utilization. The curves for the four scenarios are barely distinguishable. Scenario #3, with the largest bandwidth-delay product, is shown with solid curves, and for this we see that the link is slightly under-utilized for buffer sizes significantly smaller than the bandwidth-delay product. We can conclude that the single variable defined by $B/(CT)$ determines the bandwidth sharing between TCP

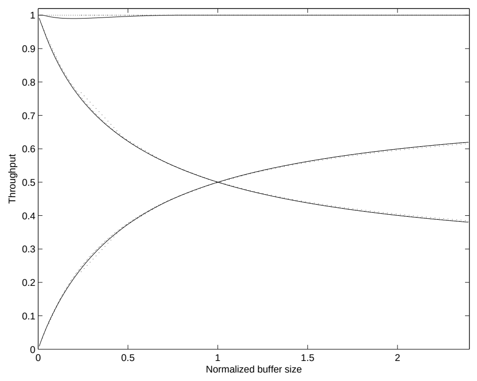

Fig. 4. Normalized throughput vs. normalized buffer size $B/(CT)$. The theoretical throughput curves for all four scenarios are plotted together.

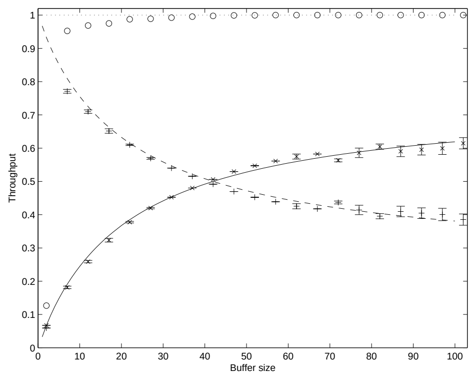

Fig. 5. Validation for scenario #1 - Normalized throughput vs. buffer size (in packets).

Westwood+ and TCP New Reno.

To validate the present model and the assumptions (such as fluid flow approach and full synchronization), we simulate the same scenarios using ns2, and compare these results to the theoretically computed values. We assign random start times to the two flows, measure the actual throughput for a long transmission, excluding transients at flow startup, and average over several realizations. For lack of space we only show the throughputs for scenario #1 in Fig. 5 with 95% confidence intervals. The results for the other three scenarios lead to the same conclusions. In Figure 5, the vertical axis corresponds to the normalized throughput and the horizontal axis corresponds to the buffer size in packets. The ns2 simulations are displayed for New Reno with marks $\times$, Westwood+ with marks $+$, and their sum with marks $\circ$. The curves show the theoretical results.

The ns2 simulation and the analysis give throughput values that match remarkably well, except for very small buffer sizes ($B < 5$). The most likely cause of the mismatch for small values of B is the fluid model approach.

Consider the ratio between Westwood+ and New Reno throughput. As can be seen in the figure, the ratio decreases as the buffer size is increased and it tends to 1/3 rather than the $1/\sqrt{2}$ that one obtains by application of the "square root formula" in [12].

For TCP Westwood+ to function properly, it needs to estimate $\text{RTT}_{\min}$. It shares this requirement with delay based congestion control methods such as TCP Vegas and Fast TCP. If a Westwood+ flow is started at a time when the bottleneck link is highly loaded, the buffer may stay close to full for the whole duration of the flow. Then, in effect, part of the queueing delay will be erroneously accounted for as propagation delay, which results in an overestimation of the real bandwidth delay product and hence in a less dramatic reduction of the congestion window. In this case, the disadvantage that Westwood+ has when competing with New Reno over large buffer, is actually reduced. This is illustrated in Fig. 6, which shows the result of a simulation of scenario #1 (c.f., Fig. 5), with the only difference being that the start of the Westwood+ flow is delayed around 50 seconds so that the TCP New Reno flow gets enough time to fill the buffer. For the larger buffer sizes, Westwood's $\text{RTT}_{\min}$ estimate has values up to 80ms, to be compared to the true round trip propagation delay of 50ms.

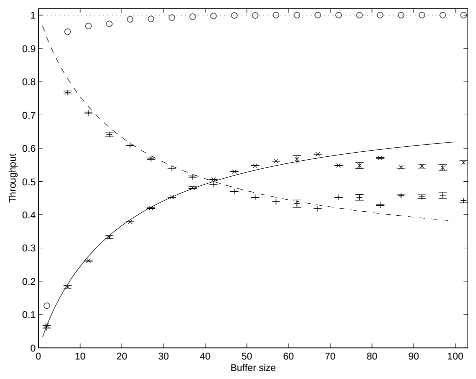

Fig. 6. This figure shows the result for scenario #1, (c.f., Fig. 5), but with the start of the Westwood+ flow delayed around 50 s.

Furthermore, when the number of flows over the bottleneck links is increased, it becomes less probable that the router buffer empties. Thus, also in this case, one can expect the $\text{RTT}_{\min}$ estimate to include some of the queueing delay. And consequently, the disadvantage of Westwood+ can be expected to be reduced when more flows are multiplexed.

To improve fairness in the large buffer case, the window decrease of Westwood+ in response to a packet loss could be modified, so that the new window is never set to a value smaller than half of the previous value. In other words, modify Westwood+ to never decrease its window more than New Reno would have done. To validate this claim, we modified the Westwood+ code in ns-2 accordingly and rerun the simulation for scenario #1. The results are presented in Fig. 7. Clearly, for buffers larger than the bandwidth delay product when it is very probable that TCP Westwood+ divides its window by more than 2, our modification solves the unfairness problem. At the same time, the performance of both protocols for buffers smaller than the bandwidth delay product stays the same.

For successful deployment of TCP Westwood+ as a general purpose congestion control mechanism over the Internet without significant negative effects on New Reno flows, it seems prudent not to set the size of buffers in routers to small values. It is known (see e.g., [3], [7]) that for small

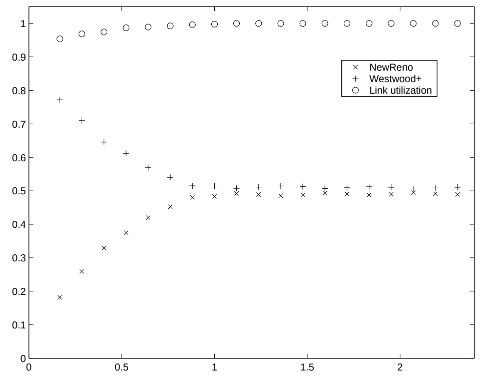

Fig. 7. The performance of TCP Westwood+ in scenario #1 after our modification - Normalized throughput vs. normalized buffer size ($B/(CT)$).

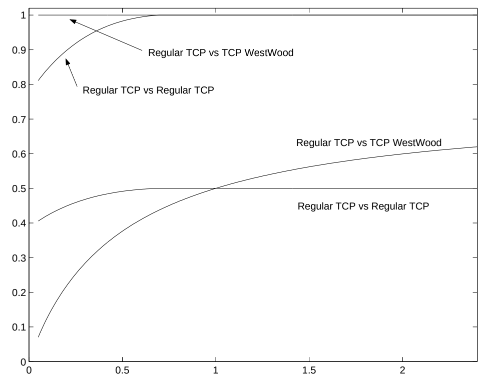

Fig. 8. The normalized throughput of TCP New Reno when competing with either TCP New Reno or TCP Westwood+ as a function of the normalized buffer size.

buffers, TCP New Reno under-utilizes the available bandwidth at the bottleneck link. In contrast, TCP Westwood+ is able to utilize all available bandwidth even when the buffers are small. One could ask a question if TCP Westwood+ takes only un-used bandwidth when it competes with TCP New Reno, or it steals this bandwidth from TCP New Reno. To answer this question, we consider two TCP New Reno flows instead of the mix of TCP Westwood+ and TCP New Reno and we vary the ratio between the buffer size and the bandwidth delay product. The results are depicted in Fig. 8. In this figure, we plot normalized throughput of TCP New Reno when the competing TCP connection is either TCP New Reno or TCP Westwood+. The two topmost curves correspond to link utilization. We can read in this figure that when two TCP New Reno flows compete for the available bandwidth and the bottleneck buffer is small, the link is not fully utilized, as expected. The figure demonstrates that TCP New Reno does suffer from the presence of TCP Westwood+ flow when the buffer size is smaller than the bandwidth-delay product. For buffers larger than the bandwidth delay product, regular TCP realizes better performances after the introduction of TCP Westwood+. Note that our modification to TCP Westwood+ introduced earlier can prohibit regular TCP from stealing bandwidth from TCP Westwood+ when buffers are large and hence, solve the unfairness problem in this region.

V. Conclusions

In this paper, we studied analytically and by the means of ns-2 simulations, the inter-protocol fairness between TCP Westwood+ and TCP New Reno. Until the present, the effect of the introduction of TCP Westwood+ on TCP New Reno was not thoroughly investigated. We explained why these protocols when they compete for available bandwidth get different shares. Our main conclusion is that the bandwidth sharing only depends on the ratio between the buffer size at the bottleneck router and the bandwidth delay product. In particular, if the ratio is smaller than one, TCP Westwood+ takes more bandwidth. On the contrary, if the ratio is greater than one, it is TCP New Reno which gets the larger part.

The introduction of TCP Westwood+ allows to solve the well known problem of network under-utilization by regular TCP when buffer sizes in routers are set to small values. Unfortunately, this gain in the utilization comes at the expense of regular TCP which loses some of its throughput.

Inspired by our results, we proposed a simple modification to TCP Westwood+ that solves the unfairness problem for large buffer sizes. For small buffers, the unfairness problem is still open.

References

[1] S. Mascolo, C. Casetti, M. Gerla, M. Y. Sanadidi, and R. Wang, "TCP Westwood: bandwidth estimation for enhanced transport over wireless links," in *MobiCom*, Rome, Italy, 2001.

[2] S. Mascolo and F. Vacirca, "Congestion control and sizing router buffers in the internet," in *IEEE Conference on Decision and Control and European Control Conference*. Seville: IEEE CSS, 2005, pp. 6750–6755.

[3] K. Avrachenkov, U. Ayesta, E. Altman, P. Nain, and C. Barakat, "The effect of router buffer size on the TCP performance," in *Proceedings of LONIIS workshop*, 2002.

[4] G. Appenzeller, I. Keslassy, and N. McKeown, "Sizing router buffers," in *SIGCOMM*. Portland: ACM, September 2004.

[5] A. Dhamdhere, H. Jiang, and C. Dovrolis, "Buffer sizing for congested internet links," in *Proceedings of IEEE INFOCOM*, 2005.

[6] S. Gorinsky, A. Kantawala, and J. Turner, "Link buffer sizing: A new look at the old problem," in *Proceedings of IEEE Symposium on Computers and Communications*, June 2005, pp. 507–514.

[7] K. Avrachenkov, U. Ayesta, and A. Piunovskiy, "Optimal choice of the buffer size in the internet routers," in *IEEE Conference on Decision and Control and European Control Conference*. Seville: IEEE CSS, 2005.

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

[9] R. M. Corless, G. H. Gonnet, D. E. H. Hare, D. J. Jeffrey, and D. E. Knuth, "On the Lambert W function," *Advances in Computational Mathematics*, vol. 5, pp. 329–359, 1996.

[10] F. Baccelli and D. Hong, "AIMD, fairness and fractal scaling of TCP traffic," in *Proceedings of IEEE INFOCOM*, New York, June 2002, pp. 229–238.

[11] A. Leizarowitz, R. Stanojevic, and R. Shorten, "Tools for the analysis and design of communication networks with markovian dynamics," *to appear in Proceedings of IEE, Control Theory and Applications*, 2005.

[12] L. A. Grieco and S. Mascolo, "Performance evaluation and comparison of Westwood+, New Reno and Vegas TCP congestion control," *ACM Computer Communication Review*, vol. 34, no. 2, April 2004.