Issues in Model-Based Flow Control

Sridhar Ramesh and Injong Rhee
Department of Computer Science
North Carolina State University
Raleigh, NC 27695-7534

Abstract

This report examines potential stability problems associated with a model based apporach to TCP-friendly flow control for non-TCP traffic. In specific, such an approach involves using a TCP-friendly formula that estimates the throughput of a TCP session with the same end-to-end traffic characteristics as the non-TCP connection under consideration. The inputs to this formula include the round trip time, the timeout value, and the packet loss fraction of the connection. This paper shows that estimating the loss fraction per transmitted packet highly depends on the current transmission rate of the connection as well as the actual loss fraction of the path. Thus the estimated loss fraction can contain errors which result in inaccurate estimation of the corresponding TCP throughput. This inaccuracy can push the transmission rate of the non-TCP connection away from the fair share of the bottleneck bandwidth on the end-to-end path, so that under steady state, the connection ends up receiving either over-allocation or under-allocation of bandwidth.

1 Introduction

Congestion control is an integral part of any best-effort Internet data transport protocol. It is widely accepted that the congestion avoidance mechanisms employed in TCP [1] have been one of the key contributors to the success of the Internet. A conforming TCP flow is expected to respond to congestion indication (e.g., packet loss) by drastically reducing its transmission rate and by slowly increasing its rate during steady state. This congestion control mechanism encourages the fair sharing of a congested link among multiple competing TCP flows. A data flow is said to be *TCP-friendly* if at steady state, it uses no more bandwidth than a conforming TCP connection running under comparable conditions.

Recently we have seen several efforts to develop a stochastic model of TCP congestion control that gives a simple analytical formula for the throughput of a TCP sender as a function of packet loss and round trip time (RTT) [2, 7, 3]. These efforts are propelled by the interests in using the formula for TCP-friendly flow control of a non-TCP flow such as UDP traffic [6, 8] and reliable multicast [5]. Typically, these flow control schemes work as follows. A receiver monitors packet loss rates and round trip delays, and using a TCP friendly formula, it estimates the throughput of a TCP connection running under the same operating conditions. The estimated throughput is sent as feedback to the sender. If the feedback throughput is less than or equal to the current transmission rate of the non-TCP flow, then the sender sets its rate to the feedback throughput. Otherwise, it increases its rate.

The stochastic model used by Padhye et al. [3] makes the TCP throughput estimation based on the following assumptions:

- When a packet is lost, all subsequent packets in the same RTT round are lost.
- The probability that a packet is lost in an RTT round, given that no previous packet in the same round is lost is independent of packet loss in earlier rounds. Call this probability l_{act}.

To measure l_{act} of a TCP flow under observation, Padhye et al.[3] counts the number of TCP loss indictions (triple duplicate acknowledgements, and timeouts) over a certain period, and divides the result by the total number of packets transmitted by TCP over that duration. This is an approximation. Let l_{app} be the expected value of the resulting value. l_{app} is used as input to their formula to estimate the throughput of a TCP connection under observation.

When a non-TCP flow is regulated using the formula, it is not possible to compute l_{app} since the flow may use a different window size. Instead, Handley et al. [5] and Padhye et al.[4] estimate l_{act} on the non-TCP flow by dividing the number of *loss events* by the total number of packets transmitted. Loss events are registered as follows. The first packet loss is counted as a loss event. Following this, there is a back-off for the duration of an RTT during which no packet loss is counted. The next packet loss after this back-off is counted as a loss event, followed by another RTT back-off, and so forth. Let the expected value of this be l_{est}.

Let B be the total bandwidth shared among n TCP sessions, and one non-TCP flow running on the same end to end path. The objective of TCP-friendly algorithms is to ensure that the rate μ of the non-TCP flow is maintained at the fair share given by $B_{fair} = \frac{B}{n+1}$, in steady-state. The known TCP-friendly algorithms achieve this by a feedback mechanism. Specifically, an estimate of the throughput on each of the n TCP sessions is fed back to the sender which then uses this to regulate its transmitting rate μ. Assuming that the TCP sessions share the residual bandwidth equally, the steady-state throughput of each TCP session is $\frac{B-\mu}{n}$. Ideally, this is the parameter which must be fed back to the sender. It can be seen that when $\mu > B_{fair}$, $\frac{B-\mu}{n} < B_{fair}$, and when $\mu < B_{fair}$, $\frac{B-\mu}{n} > B_{fair}$.

However, as discussed earlier, the TCP throughput estimate is calculated based on the loss fraction of packets on the non-TCP flow. This obviously means that errors in estimating the loss fraction could lead to errors in estimating TCP throughput. In the following sections, we show that this indeed happens, and could even result in unfair allocation of bandwidth. In particular, we show that:

1. When $\mu > \frac{1}{RTT}$ (i.e., one packet per RTT), $l_{est} < l_{act}$. As a result, by substituting l_{est} for l_{act} in any formula which estimates TCP throughput based on l_{act}, we obtain an erroneous estimate. This results in over-estimation of the throughput.

2. When $\mu < B_{fair}$, $l_{est} > l_{app}$. Thus, substituting l_{est} for l_{app} in a formula estimating TCP throughput based on l_{app} gives us an erroneous estimate. This results in over-estimation of the throughput.

3. When $\mu > B_{fair}$, we could have $l_{est} < l_{app}$ under certain conditions. Thus, substituting l_{est} for l_{app} in a formula estimating TCP throughput based on l_{app} gives us an erroneous estimate and under-estimation of TCP throughput.

4. The formula used to estimate TCP throughput, such as the one provided in [3], may itself have some inaccuracy.

5. Depending on the starting rate of μ, these errors introduced in TCP throughput estimation push μ further away from the fair share, rather than forcing it to converge to the fair share. Thus, at steady state, the non-TCP flow may end up receiving either over-allocation or under-allocation of bandwidth.

In Section 2, we briefly outline the TCP-friendly flow control algorithm proposed in [5], and the formula used in TCP throughput estimation. In Section 3, we describe the sources of error in TCP throughput estimation, and show how these errors result in unfair allocation of bandwidth by the flow control algorithm. Section 4 contains numerical examples which substantiate these findings.

2 Model-based flow control

The first TCP throughput estimation model was proposed by Floyd [2] and Ott et al.[7]. Their model gives the following formula:

$$B(l) = \frac{1.22s}{t_{RTT}\sqrt{(l)}} \tag{2.1}$$

where l is the loss fraction of the TCP packets, and t_{RTT} is the round trip delay of an end-to-end path where TCP runs.

l is defined as the probability that a packet is lost given that no previous packet in the same *round* is lost (we use l_{act} and l interchangeably). l is estimated by counting the number of *loss events* over a certain period and dividing the result by the number of packets sent.

Padhye et al. [3] proposed annother model which gives a better approximation than the earlier one. The model uses the following equation[1]:

$$B(l) = \frac{s}{t_{RTT}\sqrt{\frac{2bl}{3}} + t_0 min\left(1, 3\sqrt{\frac{3bl}{8}}\right) l(1+32l^2)} \tag{2.2}$$

where t_0 is the timeout value.

[1] This is only an approximation formula. The complete formula can be found in [3]

Let μ be the bit rate of transmission of packets over the non-TCP connection. Let B be the total bandwidth available between the source and a given destination. If there are n active TCP sessions between the source and the destination, the throughput of each, in steady state, is given by $\frac{B-\mu}{n}$.

A typical model-based flow control works as follows. Using one of the above formulae, and estimated values of l, t_{RTT}, and t_0, the receiver of the non-TCP flow computes a value of $B(l)$ which is fed back to its sender. If $B(l) \leq \mu$, the sender sets its transmission rate to $B(l)$. Otherwise, it increases its rate by some amount. The receiver continues to report a new value of $B(l)$, and the sender makes the adjustment of its rate accordingly. The idea is that if the feedback throughput value is accurate enough, the transmission rate eventually converges to the fair share of bandwidth (i.e., $\frac{B}{n+1}$).

3 Errors in TCP Throughput Estimation and their Impact on Resource Allocation

In this section, we look at the different sources that cause errors in estimating the equivalent TCP throughput and show that their potential consequence is unfair allocation of resources to the non-TCP flow.

3.1 Error in Loss Fraction Estimation

Let l_{act} be the actual loss fraction, and let l_{est} be the expected value of the estimate measured on the non-TCP connection. Let l_{app} be the expected value of an estimate measured on one of the TCP connections. Then, we have

THEOREM **1**

$$l_{est} > l_{app} \quad \text{if} \quad \mu < \frac{B-\mu}{n}$$

For proof, we compare the packets sent over the non-TCP stream over the measurement period T, to that sent over a TCP session. Let T be equal to N round trip times. Let W_i, $i = 1, 2, ..., N$, be the number of packets sent in the i^{th} round over the TCP connection under observation. Therefore, the number of packets sent over the TCP session is given by $\sum_{i=1}^{N} W_i$. The expected number of loss events is given by :

$$Loss_{TCP} = \sum_{i=1}^{N} \sum_{j=1}^{W_i} (1 - l_{act})^{j-1} l_{act}$$

Therefore, we have:

$$l_{app} = \frac{\sum_{i=1}^{N} \sum_{j=1}^{W_i} (1 - l_{act})^{j-1} l_{act}}{\sum_{i=1}^{N} W_i} \tag{3.3}$$

In the same duration, μT packets are transmitted over the non-TCP session. The expected number of loss events observed here is given by:

$$Loss_{Mcast} = \sum_{i=1}^{N} \sum_{j=1}^{T\mu/N} (1 - l_{act})^{j-1} l_{act}$$

Thus, we have:

$$l_{est} = \frac{\sum_{i=1}^{N} \sum_{j=1}^{T\mu/N} (1 - l_{act})^{j-1} l_{act}}{T\mu} = \frac{N \sum_{j=1}^{T\mu/N} (1 - l_{act})^{j-1} l_{act}}{T\mu} \tag{3.4}$$

Since $\mu < \frac{B-\mu}{n}$, and TCP is in steady state (congestion avoidance mode), there are more packets transmitted on any TCP connection, than on the non-TCP connection. Hence, $\sum_{i=1}^{N} W_i > T\mu$.

LEMMA **1** *For any q such that $0 < q < 1$, $F_m(q) = \frac{1+q+q^2+....+q^{m-1}}{m}$ is a monotonically decreasing function m.*

PROOF : $F_{m+1}(q) = \frac{1+q+q^2+....+q^{m-1}+q^m}{m+1}$. $q < 1$.
Hence, $q^m < q^{m-1} < q^{m-2} < ..... < q^2 < q < 1$.
$F_{m+1}(q)$ can be rewritten as $\frac{1+q^m/m+q+q^m/m+q^2+q^m/m....+q^{m-1}+q^{m-1}/m}{m+1}$.
Thus, we have:

$$F_{m+1}(q) < \frac{1 + 1/m + q + q/m + ... + q^{m-1} + q^{m-1}/m}{m+1} = \frac{1 + q + ... + q^{m-1}}{m} = F_m(q)$$

LEMMA **2** *If $G_m(q) = F_m(q) \times m = 1 + q + q^2 + ... + q^{m-1}$, and $M = \frac{\sum_{i=1}^{N} m_i}{N}$, then*

$$\sum_{i=1}^{N} G_{m_i}(q) \leq N \times G_M(q)$$

PROOF : Consider the sequence $\{q^{m_1}, q^{m_2}, ..., q^{m_N}\}$. Its arithmetic mean is given by $AM(q) = \frac{\sum_{i=1}^{N} q^{m_i}}{N}$. Its geometric mean is $GM(q) = q^M$, where $M = \frac{\sum_{i=1}^{N} m_i}{N}$. $AM(q) \geq GM(q)$. Also, $0 < q < 1$, so $(1-q) > 0$. Therefore, $\frac{N - AM(q)}{1-q} \leq \frac{N - GM(q)}{1-q}$. But $\frac{N - AM(q)}{1-q} = \sum_{i=1}^{N} G_{m_i}(q)$, and $\frac{N - GM(q)}{1-q} = N \times G_M(q)$.

We have:

$$l_{app} = \frac{\sum_{i=1}^{N} \sum_{j=1}^{W_i} (1 - l_{act})^{j-1} l_{act}}{\sum_{i=1}^{N} W_i} = \frac{\sum_{i=1}^{N} G_{W_i}(1 - l_{act}) \times l_{act}}{\sum_{i=1}^{N} W_i}$$

Therefore, from Lemma 2, it is seen that:

$$l_{app} \leq \frac{N \times G_{EW}(1 - l_{act}) \times l_{act}}{N \times EW} \quad \text{where} \quad EW = \frac{\sum_{i=1}^{N} W_i}{N} \tag{3.5}$$

But $N \times EW = \sum_{i=1}^{N} W_i > \mu T$. Hence, from Lemma 1, we have:

$$\frac{N \times G_{EW}(1 - l_{act}) \times l_{act}}{N \times EW} < \frac{N \times G_{(\frac{T\mu}{N})}(1 - l_{act}) \times l_{act}}{T\mu} \tag{3.6}$$

Recognize that the RHS in (3.6) is l_{est}. Thus, combining (3.6) and (3.5) gives us the proof of Theorem 1.

THEOREM **2**

$$l_{est} < l_{app} \quad \text{if} \quad \mu > \frac{B - \mu}{n}, \quad \text{and} \quad \frac{T\mu}{N} > W_i, \quad \text{for} \quad 1 \leq i \leq N$$

From Lemma 1, it follows that $\frac{G_m(q)}{m} < \frac{G_k(q)}{k}$, for $k < m$. Therefore, $k \times G_m(q) < m \times G_k(q)$. Observe that l_{est} may be rewritten as :

$$l_{est} = \frac{\sum_{i=1}^{N} l_{act} \times G_{(T\mu/N)}(1 - l_{act})}{T\mu} = l_{act} \times N \times \frac{G_{(T\mu/N)}(1 - l_{act})}{T\mu}$$

Therefore, $l_{est} \times \frac{T\mu}{N} = l_{act} \times G_{(\frac{T\mu}{N})}(1 - l_{act})$. But $\frac{T\mu}{N} > W_i$, for $1 \leq i \leq N$. Hence, for $1 \leq i \leq N$, we have:

$$G_{(\frac{T\mu}{N})}(1 - l_{act}) \times W_i < G_{W_i}(1 - l_{act}) \times \frac{T\mu}{N}$$

Taking the summation over $1 \leq i \leq N$, we get:

$$\sum_{i=1}^{N} G_{(\frac{T\mu}{N})}(1 - l_{act}) \times W_i \times l_{act} < \sum_{i=1}^{N} G_{W_i}(1 - l_{act}) \times \frac{T\mu}{N} \times l_{act} \tag{3.7}$$

Simplifying both sides, we get:

$$l_{est} \times \frac{T\mu}{N} \times \sum_{i=1}^{N} W_i < l_{app} \times \frac{T\mu}{N} \times \sum_{i=1}^{N} W_i$$

Or, $l_{est} < l_{app}$.

THEOREM **3** *When* $\mu > \frac{1}{RTT}$, $l_{est} < l_{act}$.

For proof, we revisit Lemma 1. Observe that:

$$l_{est} = \frac{\sum_{i=1}^{N} l_{act} \times G_{(T\mu/N)}(1 - l_{act})}{T\mu} = l_{act} \times \frac{G_{(T\mu/N)}(1 - l_{act})}{T\mu/N} = l_{act} \times F_{(T\mu/N)}(1 - l_{act})$$

But $\frac{T}{N} = RTT$, and $\frac{T\mu}{N} > 1$. Therefore, from Lemma 1, it follows that

$$F_{(T\mu/N)}(1 - l_{act}) < F_1(1 - l_{act}) = 1$$

Hence, $l_{est} < l_{act}$.

3.2 Impact of Errors in Loss Fraction on TCP Throughput Estimation

First, consider the case where TCP throughput is estimated as a function of l_{act}, such as (2.2). From (2.2), it can be seen that the estimate $B(l)$ is a decreasing function of l. Thus, when $l_{est} < l_{act}$, the estimate $B(l_{est})$ is likely to be larger than the actual value of $\frac{B-\mu}{n}$ which is the correct feedback parameter. In the extreme case, we may have the following situation:

$$B_{fair} < \mu < B_{feed} = B(l_{est})$$

Consider the case where the non-TCP transmission rate μ is given by $\frac{B}{n+1} + n\Delta B$, for some $\Delta B > 0$. Now, suppose we are given a function $B(l)$ which correctly estimates the throughput on a TCP connection given the value of the loss fraction of packets, l. Let $\beta(l) = -\frac{dB}{dl} > 0$, for all l such that $0 < l < 1$. For the value of μ considered, $B(l) = \frac{B-\mu}{n}$, since we have assumed that $B(l)$ correctly estimates TCP throughput. Hence,

$$B(l) = \frac{B}{n+1} - \Delta B$$

Thus, when the value of μ is greater than the *fair share*, $\frac{B}{n+1}$, the feedback parameter is less than the fair share. This facilitates reduction of the transmission rate on the non-TCP connection to preserve bandwidth fairness.

However, when the feedback parameter is calculated based on l_{est}, the estimate of l rather than the actual value l_{act}, there is no such guarantee. As shown earlier,

$$l_{est} = \frac{\sum_{i=1}^{N} \sum_{j=1}^{T\mu/N} (1 - l_{act})^{j-1} l_{act}}{T\mu}$$

where T is the period of observation and N is the number of round trip times in T. Therefore, $l_{est} = l_{act} - \Delta l < l_{act}$. The feedback parameter $B(l_{est}) = B(l_{act} - \Delta l) \approx \frac{B}{n+1} - \Delta B + \beta(l_{act})\Delta l$. If $\Delta B < \frac{\beta(l_{act})\Delta l}{n+1}$, then we have:

$$B(l_{est}) \approx \frac{B}{n+1} - \Delta B + \beta(l_{act})\Delta l > \frac{B}{n+1} + n\Delta B = \mu$$

Next, we consider the case where TCP throughput is calculated using a formula based on l_{app}. Assuming we are provided a formula $B(.)$ that accurately estimates TCP throughput, we should have:

$$B(l_{app}) = \frac{B - \mu}{n}$$

Let $\mu = \frac{B}{n+1} + n\Delta B$. But since we use l_{est} instead of l_{app}, the feedback parameter is given by:

$$B(l_{est}) \approx \frac{B}{n+1} - \Delta B + \beta(l_{app})(l_{app} - l_{est})$$

Thus, the following problems could be encountered:

1. When $\mu < \frac{B-\mu}{n}$, $l_{est} > l_{app}$. There could be a situation where $B(l_{est}) \leq \mu < B_{fair}$.

2. When $\mu > \frac{B-\mu}{n}$, and $l_{est} < l_{app}$, there could be a situation where $B(l_{est}) \geq \mu > B_{fair}$.

3.3 Errors in TCP Throughput Model

In the earlier section, we considered the error due to inaccurate estimation of the loss fraction, assuming that the throughput model to be perfect. However, it is seen that the throughput estimation based on (2.2) also introduces some error (see [3]). From the tables provided in [3], it was observed that the relative error approached 100% in certain extreme cases. In this section, we show that this error can add on to the error in loss fraction estimation to result in erroneous feedback to the non-TCP sender.

Suppose the model estimates TCP's throughput to be $B_{mod}(l) = B_{act} + B_{\epsilon}$, where B_{act} is the actual TCP throughput.

Consider a link between two nodes with a minimum round trip time of T_1 seconds. Let $N_1 = [B_{act} \times T_1]$. Let $l_1 = l - \Delta l = l \times \frac{1-(1-l)^{N_1}}{N_1}$. It is easily seen that $\Delta l > 0$ if $N_1 > 1$. Let the bandwidth of the link under consideration be $B_{tot} = nB_{act} + \frac{B_{mod}(l)+B_{mod}(l-\Delta l)}{2}$. Now, suppose we have a flow of rate $\mu = \frac{B_{mod}(l)+B_{mod}(l-\Delta l)}{2}$, sharing this link with n , $n \geq 1$, TCP connections. The throughput of each TCP session is $\frac{B_{tot}-\mu}{n} = B_{act}$. Using (2.2) however, we get different results. If l is estimated correctly, the TCP bandwidth is estimated as:

$$B_{mod}(l) = B_{act} + B_{\epsilon}$$

Since $\Delta l > 0$, and $B_{mod}(l)$ is a decreasing function of l, $B_{mod}(l - \Delta l) > B_{mod}(l)$. So,

$$\mu - B_{mod}(l) = \frac{B_{mod}(l - \Delta l) - B_{mod}(l)}{2} > 0$$

In other words, $\mu > B_{act} + B_{\epsilon}$.

But l is not correctly estimated. The estimate of l is given by:

$$l_{est} = \frac{1 - (1 - l)^{\mu \times RTT}}{\mu \times RTT}$$

where RTT is the mean round-trip time. Obviously, $RTT \geq T_1$. Besides, $\mu > B_{act} + B_{\epsilon}$. Hence, from Lemma 1, it can be seen that $l_{est} < l_1 < l$. We can write $l_{est} = l - Dl$ where $Dl > \Delta l$.

The estimate of the TCP bandwidth, using (2.2), is given by:

$$B_{mod}(l_{est}) = B_{mod}(l - Dl)$$

As mentioned earlier, $B_{mod}(l)$ is a decreasing function. Therefore,

$$B_{mod}(l_{est}) = B_{mod}(l - Dl) > B_{mod}(l - \Delta l)$$

Also,

$$B_{mod}(l_{est}) - \mu > B_{mod}(l - \Delta l) - \mu = \frac{B_{mod}(l - \Delta l) + B_{mod}(l)}{2} > 0$$

The fair share is given by:

$$B_{fair} = \frac{B_{tot}}{n+1} = B_{act} + \frac{B_{mod}(l - \Delta l) - B_{mod}(l)}{2n+2} < B_{act} + B_{\epsilon} + \frac{B_{mod}(l - \Delta l) - B_{mod}(l)}{2} = \mu$$

Thus, we have a situation where $B_{mod}(l_{est}) > \mu > B_{mod}(l) > B_{fair}$, i.e., one where the error in estimating l, coupled with the error introduced by the formula (2.2) results in erroneous throughput estimation. In the next section, we show how this may even result in unfair allocation of resources.

3.4 Non-Convergence to Fair-Share

In the earlier sections, we have shown that the errors in estimating the loss fraction and errors in the TCP throughput formula could result in the following situations:

$$(i) \quad B_{feed} > \mu > B_{fair} \quad \text{OR} \quad (ii) \quad B_{feed} < \mu < B_{fair}$$

where μ is the rate of the non-TCP flow, B_{fair} is the fair share, and $B_{feed} = B(l_{est})$ is the feedback parameter.

In this section, we show that when $B_{feed} >$ (resp $<$)$\mu >$ (resp $<$)B_{fair}, the non-TCP rate never converges to the fair share in steady state. In addition, the long-term average throughput of the non-TCP flow does not converge to the fair share either.

We prove this for the case where $B_{feed} > \mu > B_{fair}$, and the proof for the case $B_{feed} < \mu < B_{fair}$ is similar. We first observe that $B_{feed} = B(l_{est})$, where l_{est} is itself a function of μ. Therefore, we can write $B_{feed} = B^{\Omega}(\mu)$. Assuming $B(l_{est})$ is a continuous function, and l_{est} is a continuous function of μ, $B^{\Omega}(\mu)$ is also continuous. We know that $B^{\Omega}(\mu) > \mu$. Hence, we have either:

$$B^{\Omega}(\nu) = \nu \quad \text{for some} \quad \nu > \mu \tag{3.8}$$

Or:

$$B^{\Omega}(\nu) > \nu \quad \text{for all} \quad \nu > \mu \tag{3.9}$$

If (3.8) is true, then the feedback parameter when the initial rate of the non-TCP flow is ν is given by $B^{\Omega}(\nu) = \nu$. Therefore, the rate as determined by the non-TCP flow control protocol converges to $\nu > \mu > B_{fair}$. The long-term average of the flow is also ν.

If (3.9) is true, when the initial rate ν_0 is greater than μ, it can be seen that the feedback parameter $B^{\Omega}(\nu_0)$ is greater than ν_0. Writing $\nu_1 = B^{\Omega}(\nu_0)$, and in general, $\nu_{i+1} = B^{\Omega}(\nu_i)$ for $i > 0$, we have:

$$B^{\Omega}(\nu_i) > \nu_i \geq \nu_0$$

In practice, $\nu_{i+1} = min\left(B^{\Omega}(\nu_i), B_{max}\right)$, but since $\nu_i \leq B_{max}$, we can still write:

$$\nu_{i+1} \geq \nu_i \geq \nu_0$$

This means $\nu_i > \mu > B_{fair}$, for all $i \geq 0$. The long-term average of the flow is given by:

$$\gamma_\infty = \lim_{N\to\infty} \frac{\sum_{i=0}^{N-1} \nu_i}{N} > \lim_{N\to\infty} \frac{\sum_{i=0}^{N-1} \mu}{N} = \mu > B_{fair}$$

4 Numerical Examples

We consider some numerical examples of cases where the flow control algorithm using formula based feedback results in unfair allocation of resources.

In the first example, the formula is assumed to accurately compute the TCP throughput when the loss rate is correctly estimated, and the error in TCP throughput estimation is due to erroneous estimation of the loss fraction. There are two TCP sessions sharing the bottleneck link which has a bandwidth B of 500 KB/s. The fair share is therefore 166.7 KB/s. We assume that the non-TCP flow has an initial rate μ of 190 KB/s. An ns simulation was accordingly set-up, with two TCP connections sharing a 4 Mb/s link with a constant bit-rate source sending packets at 190 KB/s. The observed loss fraction in the simulation is 0.0156. However, the loss fraction as estimated by dividing the number of loss events by the total number of packets transmitted is 0.0125. Using the same simulation set-up, it was determined that a loss fraction of .0125 corresponds to an actual transmitting rate μ of 120 KB/s by the constant bit-rate source. Therefore, any correct formula estimates the TCP rate as $\frac{B-\mu}{n} = \frac{500-120}{2} = 190$. In other words, the feedback parameter B_{feed} is equal to μ though μ is significantly larger than the fair share. Since $\mu = 190$, this leaves less than 155 KB/s for either of the TCP connections. Thus, the bandwidth allocation to μ is at least 23% greater than that for each TCP connection.

In the second example, we illustrate the effects of estimating the TCP throughput based on the formula given in (2.2). There could be an error introduced by the formula, compounded by an error due to inaccurate estimation of the loss fraction. Among the measurements provided in [3], we consider the case where 100 serially initiated TCP connections were established for 100 second intervals between two hosts. An average throughput of 17.13 KB/s was observed per connection, with a loss fraction of 0.0078, mean RTT of 0.2501 seconds and time out period of 2.5127. The throughput estimation, as given by (2.2) was 33.4 KB/s. Now, suppose the bottleneck is a 496 KB/s ($\approx$ 4 Mbps) link shared with 27 TCP sessions. The fair share is 17.75 KB/s, but when $\mu = 33.5$ KB/s, the TCP throughput is 17.13 KB/s. Assuming the TO and RTT parameters are the same as in the measurement described above, the approximate loss fraction l_{app} is .0078, and the formula in (2.2) estimates the TCP throughput as 33.4 KB/s. Assuming $l_{act} \approx l_{app}$, and applying the transformation for l_{est} given in (3.4), the estimate of the loss fraction on the non-TCP connection is .0076. Substituting this value for l in equation (2.2), $B_{feed} = B(l)$ is 34.01 KB/s which is higher than μ although μ itself is higher than the fair share. Thus, the resource allocation to μ is atleast about 89% higher than the fair share.

References

[1] V. Jacobson, "Congestion Avoidance and Control," *Proceedings of ACM SIGCOMM'88*, August 1988.

[2] S. Floyd, " Connections with Multiple Congested Gateways in Packet-Switched Networks Part 1: One-way Traffic," *Computer Communications Review*, vol. 21, no. 5, October 1991.

[3] J. Padhye, V. Firioiu, D. Towsley, and J. Kurose, "Modelling TCP Throughput: A simple model and its empirical validation," *Proceedings of SIGCOMM '98*, 1998.

[4] J. Padhye, J. Kurose, D. Towsley, and R. Koodli, "An Empirical Study of Client Interactions with a Continuous-Media Courseware Server," *IEEE Internet Computing*, April 1999.

[5] M. Handley, and S. Floyd, "Strawman Specification for TCP Friendly (Reliable) Multicast Congestion Control (TFMCC), *Reliable Multicast Research Group,* December 1998.

[6] J. Mahdavi, and S. Floyd, "TCP-Friendly Unicast Rate-Based Flow Control," *Techical Note on End2end Mailing List,* January 1997.

[7] T. Ott, J. Kemperman, and M. Mathis, "Window Size Behavior in TCP/IP with Constant Loss Probability," *DIMACS Workshop on Performance of Realtime Applications on the Internet,* Plainfield NJ, November 1996.

[8] D. Sisalem, F. Emanuel, and H. Schulzrinne, "The Direct Adjustment Algorithm: ATCP-Friendly Adaptation Scheme," *Preprint*, Columbia University, 1998.