ANALYTIC MODELS FOR HIGHSPEED TCP FAIRNESS ANALYSIS

Damien Phillips and Jiankun Hu
School of Computer Science and Information Technology, RMIT University
GPO Box 2476V, Melbourne 3001, Australia
Email: {damphil, jiankun}@cs.rmit.edu.au

***Abstract*— Highspeed TCP (HSTCP) has been proposed to make better use of large congestion window flows. Due to the nature of HSTCPs congestion avoidance increase rate, mathematical analysis of HSTCP fairness is rarely found in literature. We develop analytic models for HSTCP congestion avoidance. Based on the work for standard TCP fairness analysis in [1], we construct a model to predict throughput share of two long lived HSTCP flows sharing an Adaptive RED buffer, with increasingly disproportionate RTTs. We show that HSTCP inherits standard TCPs problem of fairness of flows with different RTT, even in the ideal case of non-synchronised loss. We also show that in the presence of some degree of loss synchronisation, HSTCPs fairness can degrade rapidly as the ratio of RTT increases. However, we also show that, as the shared buffer's bandwidth increases, Adaptive RED is better capable of preventing loss synchronisation.**

***Index Terms*— Modeling, Transport protocols.**

I. INTRODUCTION

With increasing bandwidth of networks, increased distribution and volume of information in the Internet comes the need to scale the Internet's main transport protocol, TCP. To reach this goal, researchers have proposed several protocols aimed at high bandwidth and high congestion window transfers. Such example protocols include, Highspeed TCP, FAST TCP and Scalable TCP [2], [3], [4]. Each of these protocols shows promise when it comes to making better use of high bandwidth-delay links. Highspeed TCP (HSTCP) [2] deserves special attention as it is engineered to behave along a similar response function as standard TCP, but scaled up to meet the demands of higher window size at higher path loss probability.

A problem for standard TCP that still remains unresolved in today's infrastructure, whether it be Droptail or Active Queue Management (AQM) buffers, is that TCP has a bias towards connections with shorter RTT. [5] This stems from the rate of increase in slow start and congestion avoidance. This rate is directly related to RTT, such that the shorter the RTT, the higher the increase rate, and hence the more bandwidth obtained, when operating in an environment with connections of longer RTT. It begs to ask, therefore, whether a high bandwidth protocol such as HSTCP, is able to improve upon the fairness between flows of different RTT that standard TCP exhibits, whether it has similar fairness properties to standard TCP, or whether it makes the problem more severe.

For standard TCP, models exist that specifically identify the features of competing flows with different RTT, over several types of shared queue/path. For the case of a simple Droptail queueing mechanism, [5] provides a result that assumes that losses are synchronised. Furthering this work, [1] suggests that AQM schemes such as RED are able to alleviate the problem of synchronised loss, and presents a model that assumes no synchronisation.

Although there exists many analytical models for standard TCP congestion analysis [5], [6], [7] , mathematical analysis of HSTCP RTT fairness is rarely available due to the nonlinear behavour of HSTCPs congestion avoidance increase rate.

In this paper, we extend the work in [1] to analyse the RTT fairness of HSTCP. We also analyse fairness in an environment where the shared path is governed by an Adaptive RED queueing mechanism. [8] We provide new models for the increase rate of HSTCP congestion avoidance. We relax the constraint of tight control over the RED queue length, such that we can model large scale scenarios with realistic queue lengths. We also relax the constraint of strict non-synchronisation, by introducing a control over the extent of loss synchronisation between flows. We then proceed to show that in an ideal case of non synchronised losses, HSTCP is able to gain improved throughput over standard TCP while displaying equal RTT unfairness as standard TCP. However, we show that with moderate levels of loss synchronisation the relative throughput of two connections of different RTT diverge quite quickly. Though we also show, through simulation, that Adaptive RED queues can manage to lower the likelyhood of synchronised losses at much larger windows, it remains a concern that in environments without AQM capability, RTT fairness problems will be amplified.

In the next section, we develop our models for HSTCP congestion avoidance. In section 3, we introduce our HSTCP extensions to the work from [1]. Section 4 provides numerical results and comparison with standard TCP results, while in Section 5 we present simulation results to validate our model's results. Section 6 follows with a conclusion of our work.

II. MODELLING HSTCP CONGESTION AVOIDANCE

Highspeed TCP has been proposed to allow TCP to make better use of high bandwidth-delay-product paths. This is achieved by modifying the response curve to cater for higher window sizes at realistic line loss rates. HSTCP defines three parameters, Low_Window, $High\_Window$ and $High\_P$, while Low_P is obtained from standard TCP's

0-7803-8783-X/04/$20.00 © 2004 IEEE

curve at Low_Window. Given HSTCP's suggested values of $Low\_Window = 38$, $High\_Window = 83000$ and $High\_P = 10^{-7}$, the following response function is obtained [2]:

$$W = 0.12/p^{0.835} \quad (1)$$

HSTCP then proceeds to formulate equations for the AIMD parameters $a(w)$ and $b(w)$. For standard TCP $a(w) = 1$ and $b(w) = 0.5$ regardless of window size w. HSTCP proposes to change both of these parameters, resulting in[1]

$$\begin{aligned} b(w) &= (H\_D - 0.5)\frac{ln(w) - ln(W)}{ln(W_1) - ln(W)} + 0.5 && (2) \\ a(w) &= w^2 * p(w) * 2 * b(w)/(2 - b(w)) && (3) \\ p(w) &= 0.078/w^{1.2} && (4) \end{aligned}$$

where $H\_D = High\_Decrease = b(High\_Window) = 0.1$, $W = Low_Window$ and $W_1 = HighWindow$.

In [1], the authors introduce a fluid model for standard TCP congestion avoidance which models a congestion window increase rate of 1 packet per RTT. This is not suitable for HSTCPs increase rate of $a(w)$. In creating a model for HSTCP congestion avoidance, we require a function to predict the window size w after a given number of RTT $n = t/RTT$. For this goal, we utilize a similar fluid model of window evolution, such that;

$$\frac{dW(t)}{dt} = \frac{dW(t)}{dack} \times \frac{dack}{dt} = \frac{a(W(t))}{W(t)} \times \frac{W(t)}{RTT} = \frac{a(W(t))}{RTT} \quad (5)$$

This forms a differential equation for which we wish to solve for $w(n)$;

$$\int \frac{1}{a(w)} dw = \int \frac{1}{RTT} dt \quad (6)$$

However, in its evaluated form, the reciprocal of $a(w)$ integrate into a function that cannot be inverted to find $w(n)$. Therefore, we formulated an expression $am(w)$ of similar structure to $a(w)$ with parameters $\{c1, c2, c3, c4, c5, c6\}$, that had an elementary integral which could be inverted to find $w(n)$. In order to find the best approximation, we constructed the following minimax problem

$$am(w) = 0.156w\frac{(c1 - c2ln(w))(c3 - c4ln(w))}{c5 - c6ln(w)} \quad (7)$$

$$\begin{aligned} \int \frac{1}{am(w)} dw &= -a \times ln(b - ln(w)) \\ &\quad + c \times ln(d - ln(w)) && (8) \\ err_{max} &= \min\left[\max\left[\frac{a(w) - am(w)}{a(w)}, \forall w\right]\right] && (9) \end{aligned}$$

To ensure that we could invert the result to obtain $w(n)$, we added a constraint to the minimisation problem such that the ratio of $c : a$ is a simple fraction. This leaves an equation to

[1] Note that the specification uses log base 10 in $b(w)$. The ratio of two log base 10 values in $b(w)$, permits our equivalent use of natural log (ln), as it simplifies analysis in resultant models.

TABLE I

COMPARISON OF W(N) TO VALUES PUBLISHED IN RFC3649.

$RTT = n$	RFC3649	$w(n)$	$RTT = n$	RFC3649	$w(n)$
100	131	134	1100	21455	22768
200	475	472	1200	25893	27613
300	1131	1117	1300	30701	32856
400	2160	2143	1400	35856	38459
500	3601	3609	1500	41336	44382
600	5477	5553	1600	47115	50585
700	7799	7998	1700	53170	57028
800	10567	10953	1800	59477	63670
900	13774	14411	1900	66013	70473
1000	17409	18358	2000	72754	77400

solve a low order polynomial fraction on $ln(w)$. The results are

$$\begin{aligned} n'(w) &= \int \frac{1}{am(w)} dw \\ &= -1985ln(12.68 - ln(w)) \\ &\quad +3970ln(25.51 - ln(w)) && (10) \\ w'(n) &= e^{25.51 - 26.46\left(f(n) - \sqrt{f(n)(-0.97 + f(n))}\right)} && (11) \\ f(n) &= e^{0.000504n} \end{aligned}$$

To complete the model we combine these results with standard TCP's congestion avoidance increase of 1 per RTT for $w < 38$. The final results become

$$n(w) = \begin{cases} w & w < 38 \\ n'(w) - n'(38) + 38 & \text{otherwise} \end{cases} \quad (12)$$

$$w(n) = \begin{cases} n & n < 38 \\ w'(n - 38) & \text{otherwise} \end{cases} \quad (13)$$

Table I, which shows a comparison of our model to values published in [2], reveals that our model follows the published values, for the target window range of HSTCP, within a 10% relative error. However, since in practice the number of RTTs between congestion events is rarely more than 400 (see table 3 in [2]), the errors in our model are normally much less.

III. MODELLING HSTCP RTT FAIRNESS

In [1], the authors present a mathematical model to analyse the RTT fairness of two standard TCP flows using a semi-Markov process with cost functions, assuming that losses are not synchronised between flows. The authors propose that two TCP sources 1 and 2 share a path of bandwidth μ, each with RTT of T_1 and T_2 respectively. Once the sum of the sources' bandwidths reach μ, a loss occurs for one connection only, and proceed to create a probability transition matrix from window size i to j. Some important assumptions are made of the system;

1) RTT of each connection is static
2) The probability of a loss for one of the connections is proportional to its bandwidth share
3) Queueing delay is short enough to be ignored, such as using a RED router to keep average delay small
4) Each loss indication is contrained to one connection at a time (whether a single packet loss is assumed is not stated).

We follow the process of [1] for our analysis of HSTCP, using the models built in the previous section. However, we state that some of the above assumptions do not hold for use in large window size TCP transfers as is targeted by HSTCP. We keep assumptions 1 and 2 as above, but alter assumptions 3 and 4. Firstly, we note that for large window size TCP flows, a significant queue length is needed to keep utilisation high and not susceptible to loss from bursty flows. Secondly, we assume that more than one packet is lost in each loss indication, and that these packets may be distributed among flows subject to assumption 2. These allow for more realistic queue sizes and for multiple packet drops per congestion event common in real networks.

A. Two HSTCP flows sharing an Adaptive RED buffer

Adaptive RED is a proposal to make the settings of thresholds and probabilities of a RED router self configurable [8]. The aim is for the administrator to decide upon an average queueing delay, so that Adaptive RED can self configure parameters such as minimum threshold, maximum threshold and associated probabilities, such that a queue length is maintained that indeed has the specified average delay. In our modelled system and simulation, this average queueing delay is set to 5ms. Therefore, for links of bandwidth of 1.5Mbps, 15Mbps and 100Mbps, average queue lengths in an Adaptive RED buffer would be 10, 16, 108. Furthermore, these are average queue lengths, such that the instantaneous queue length must exceed this substantially to cause a loss. Hence, when the sum of the two flows' bandwidths reach the shared bandwidth μ, the queue length builds up. It is this portion of time in each flow where the utilisation reaches maximum and is maintained until loss.

To model this behaviour, we consider that the shared bandwidth μ has some extra capacity $\Delta\mu$, that accounts for this queue space, and period of maximum utilisation.

From Adaptive RED, the probability of dropping increases when the average queue length is between min_th and max_th and dropped with certainty once past this threshold. [8] Considering that the average queue size does not immediatly track the instantaneous queue size, we propose that the extra capacity $\Delta\mu$ accounts for twice max_th, such that a loss will certainly occur at $\mu + \Delta\mu$. Furthermore, since loss is detected by the sender at least one RTT later, we propose that such a loss consists of multiple packet drops D, that can be shared between the two connections.

These form the modified assumptions

1) RTT of each connection is static
2) The probability of a loss for one of the connections is proportional to its bandwidth share
3) Queue sizes must be significant, as is the time taken to build up to such a queue size. Therefore, we keep assumtion 1 but adjust μ to account for the extra time at maximum utilisation
4) Each loss indication consists of D packet drops which can be distributed among the two flows

IV. Throughput Model

The next step involved in extending [1] for HSTCP, is to formulate equations for the window size of connection 1 from one congestion event, i, to the next congestion event, j, and time between two consecutive congestion events, τ_{ij}. In [1], two sets of equations are given for standard TCP, one for a loss indication on connection 1, and one for a loss indication on connection 2. For our model, we introduce a third case, whereby losses are seen on both connections. This case aims to model a synchronised loss, commonly seen in traces of two competing long lived flows.

A. Times and windows between consecutive losses

Due to the nature of (10) and (11), finding the exact point of equal time between losses where the sum of bandwidths equals μ is nontrivial. To approach this problem, we consider the available bandwidth created by connections dropping their windows, X_{av}. We then derive the time taken by each flow to individually recover X_{av} completely without influence of a competing connection. Taking the shortest time, we observe how much bandwidth the slower connection recovers individually X_{sl}, in this exact time. We propose that the ratio $X_{sl} : X_{av}$ gives a good indication of the ratio of $X_1 : X_2$ for $X_1 + X_2 = \mu$, since the ratio of bandwidth gains over this short period of time does not change significantly.

1) Case 1: Connection 1 only reacts to loss: The bandwidth made available by connection 1 reacting to loss is given by

$$X_{av} = i \times b(i)/T_1 \tag{14}$$

The times taken by each connection to individually recover this bandwidth are given by

$$\begin{aligned} \hat{i} &= \left(\mu - \frac{i}{T_1}\right) T_2 && (15)\\ t_{av11} &= \left[n(i) - n\left(i\left(1 - b(i)\right)\right)\right] T_1 && (16)\\ t_{av12} &= \left[n\left(\hat{i} + X_{av}T_2\right) - n\left(\hat{i}\right)\right] T_2 && (17) \end{aligned}$$

The percentage of bandwidth share obtained by connection 1 is then given by

$$\begin{aligned} X_{sl11} &= \left[w\left(t_{av11}/T_2 + n\left(\hat{i}\right)\right) - \hat{i}\right] / T_2 \\ X_{sl12} &= \left[w\left(t_{av12}/T_1 + n\left(i\left(1 - b(i)\right)\right)\right) \right.\\ &\quad \left. - i\left(1 - b(i)\right)\right] / T_1 \\ bs &= \begin{cases} X_{av}/\left(X_{av} + X_{sl11}\right) & t_{av11} < t_{av12} \\ X_{sl12}/\left(X_{av} + X_{sl12}\right) & \text{otherwise} \end{cases} \qquad (18) \end{aligned}$$

The window at the end of the congestion avoidance period, and the time taken become

$$\begin{aligned} g(i) &= bs \times X_{av} \times T_1 + i(1 - b(i)) && (19)\\ \tau_{ig(i)} &= \left[n(g(i)) - n(i(1 - b(i)))\right] T_1 && (20) \end{aligned}$$

2) Case 2: Connection 2 only reacts to loss: The bandwidth made available by connection 2 reacting to loss is given by

$$X_{av} = \hat{i} \times b(\hat{i})/T_2 \quad (21)$$

The times taken by each connection to individually recover this bandwidth are given by

$$t_{av21} = [n(i + X_{av}T_1) - n(i)]\, T_1 \quad (22)$$
$$t_{av22} = \left[n\left(\hat{i}\right) - n\left(\hat{i}\left(1 - b(\hat{i})\right)\right)\right] T_2 \quad (23)$$

The percentage of bandwidth share obtained by connection 1 is then given by

$$X_{sl21} = \left[w\left(t_{av21}/T_2 + n\left(\hat{i}\left(1 - b(\hat{i})\right)\right)\right) - \hat{i}\left(1 - b(\hat{i})\right)\right] / T_2$$
$$X_{sl22} = [w(t_{av22}/T_1 + n(i)) - i]\,/T_1$$
$$bs = \begin{cases} X_{av}/(X_{av} + X_{sl21}) & t_{av21} < t_{av22} \\ X_{sl22}/(X_{av} + X_{sl22}) & \text{otherwise} \end{cases} \quad (24)$$

The window at the end of the congestion avoidance period, and the time taken become

$$\hat{g}(i) = bs \times X_{av} \times T_1 + i \quad (25)$$
$$\tau_{i\hat{g}(i)} = [n(\hat{g}(i)) - n(i)]\, T_1 \quad (26)$$

3) Case 3: Both connections react to loss: The bandwidth made available by both connections reacting to loss is given by

$$X_{av} = i \times b(i)/T_1 + \hat{i} \times b(\hat{i})/T_2 \quad (27)$$

The times taken by each connection to individually recover this bandwidth are given by

$$t_{av31} = [n(i + X_{av}T_1) - n(i(1 - b(i)))]\, T_2 \quad (28)$$
$$t_{av32} = \left[n\left(\hat{i} + X_{av}T_2\right) - n\left(\hat{i}\left(1 - b(\hat{i})\right)\right)\right] T_2 \quad (29)$$

The percentage of bandwidth share obtained by connection 1 is then given by

$$X_{sl31} = \left[w\left(t_{av31}/T_2 + n\left(\hat{i}\left(1 - b(\hat{i})\right)\right)\right) - \hat{i}\left(1 - b(\hat{i})\right)\right] / T_2$$
$$X_{sl32} = [w(t_{av32}/T_1 + n(i(1 - b(i)))) - i(1 - b(i))]\,/T_1$$
$$bs = \begin{cases} X_{av}/(X_{av} + X_{sl31}) & t_{av31} < t_{av32} \\ X_{sl32}/(X_{av} + X_{sl32}) & \text{otherwise} \end{cases} \quad (30)$$

The window at the end of the congestion avoidance period, and the time taken become

$$gs(i) = bs \times X_{av} \times T_1 + i(1 - b(i)) \quad (31)$$
$$\tau_{igs(i)} = [n(gs(i)) - n(i(1 - b(i)))]\, T_1 \quad (32)$$

B. Window size stationary distribution

As in [1], we define a state space for the possible window sizes i of connection 1, $\chi \subset 1, W_1^{MAX}, i \in \chi$, and construct a Markov chain and associated probability transition matrix P. However, we include in P the third case of synchronised loss. Our transition matrix $P = p_{ij \in \chi}$, is described as follows,

$$p_1 = \frac{1}{\mu T_1}$$
$$p_2 = 1 - p_1$$
$$p_{ij} = \begin{cases} 0 & \\ +p_1^D & \text{if } j = g(i) \\ +p_2^D & \text{if } j = \hat{g}(i) \\ +1 - (p_1^D + p_2^D) & \text{if } j = gs(i) \end{cases} \quad (33)$$

where D is the number of packet drops in each congestion event.

We then use this matrix P to obtain the stationary distribution of the Markov chain $\pi = (\pi_i)_{i \in \chi}$. This can be achieved by transforming the transistion maxtrix P into an arrival/departure Q matrix and solving the set of equations subject to a total probability of 1.

$$Q = P^T - I \quad (34)$$
$$\pi Q = 0 \quad (35)$$
$$\sum_{i \in \chi} \pi_i = 1 \quad (36)$$

C. Cost functions of Markov Process

The stationary distribution $\pi = (\pi_i)_{i \in \chi}$ describes the probability that connection 1 experiences a loss at window size i. As in [1], we must define cost functions to enable calculation of expected throughput. We define f_{ij} as the total number of packets sent while in state i, for a time of τ_{ij} seconds, and formulate f_i and τ_i averaged over the three possible loss scenarios

$$f_i = f_{ig(i)}p_{ig(i)} + f_{i\hat{g}(i)}p_{i\hat{g}(i)} + f_{igs(i)}p_{igs(i)}$$
$$= p_1^D \int_{i(1-b(i))}^{g(i)} w(n) + p_2^D \int_{i}^{\hat{g}(i)} w(n) + (1 - (p_1^D + p_2^D)) \int_{i(1-b(i))}^{gs(i)} w(n) \quad (37)$$
$$\tau_i = p_1^D\,[n(g(i)) - n(i(1 - b(i)))]\,/T_1 + p_2^D\,[n(\hat{g}(i)) - n(i)]\,/T_1 + (1 - (p_1^D + p_2^D))\,[n(gs(i)) - n(i(1 - b(i)))]\,/T_1 \quad (38)$$

D. Calculation of throughput

With the constructed probability matrix P, and cost functions f_i and τ_i defined for HSTCP, the calculation of throughput for connection 1 is equivalent to that in [1], repeated here for completeness

$$\bar{X}_1 = \frac{\sum_{i \in \chi} \pi_i f_i}{\sum_{i \in \chi} \pi_i \tau_i} \quad (39)$$

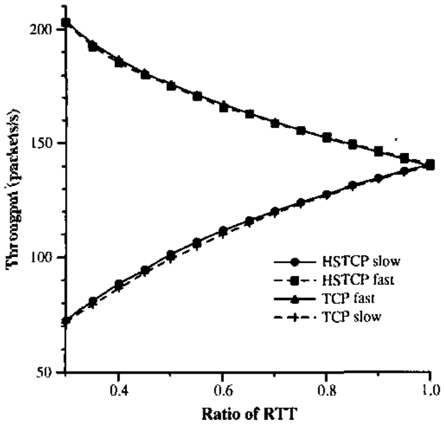

Fig. 1. Numerical comparison of RTT fairness of HSTCP and standard TCP. Bandwidth = 1.5Mbps, packet size = 576bytes, slow RTT = 0.5 seconds.

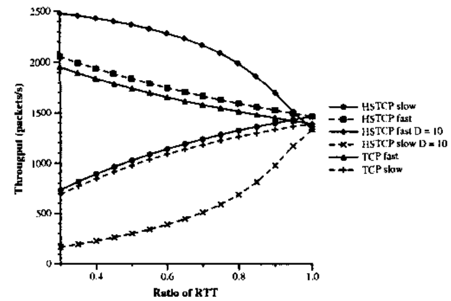

Fig. 2. Numerical comparison of RTT fairness of HSTCP, synchronised loss HSTCP and standard TCP. Bandwidth = 15Mbps, packet size = 576bytes, slow RTT = 0.5 seconds.

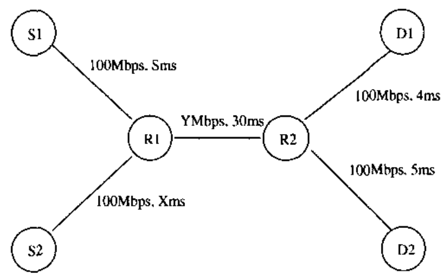

Fig. 3. Simulation topology. S1 connects to D1, S2 to D2. 5ms is part of the fixed longer RTT connection (slow). Xms is part of the variable shorter RTT connection (fast). YMbps defines the bottleneck bandwidth.

V. Numerical Results

A. Model parameters

Our model requires four main input parameters, T_1, T_2, μ and D. The round trip times T_1 and T_2 are equivalent in use to [1], while μ, the bandwidth of the shared connection, requires adjustment, and D, the number of packets dropped per congestion event is a new parameter.

As discussed earlier, we propose adding an extra capacity, $\Delta\mu$ to μ, to account for the full utilisation when the RED queue builds up. Given a path bandwidth B in bps, a packet size p, and an average Adaptive RED queue delay q, this extra capacity is calculated as

$$\Delta\mu = \min\left[5, \frac{B \times q}{8 \times 2 \times p}\right] \times 6 \tag{40}$$

The number of packets dropped per congestion event D, is designed to model flows that do not suffer strict synchronised losses, nor the simplified case of losses contrained to one connection at a time. However, $D = 1$ is equivalent to the strictly non-synchronised loss assumption, and $D \to \infty$ is equivalent to the strictly synchronised loss assumption.

B. Comparison of HSTCP to standard TCP

We solved numerically our model for HSTCP, using the same scenario as in [1], and compared it to the results for standard TCP using the same parameters. Figure 1 shows the comparison of HSTCP to TCP with $\mu \equiv 1.5$Mbps, T_1 fixed at 0.5 seconds, T_2 ranging from 0.15 to 0.5 seconds, and $D = 1$. We then repeated both evaluations with $\mu \equiv 15$Mbps, and added a third set of results to show a scenario of moderate synchronisation of loss, $D = 10$.

As can be seen in Figure (1), the comparison of HSTCP to standard is very similar with slightly greater throughput to HSTCP. This makes sense as HSTCP only begins to change the increase rate of congestion avoidance for windows greater than 38, while windows in this scenario rarely push above 80. Figure (2) shows the same comparsion but at 15Mbps. We can see that HSTCP has throughput gains over standard TCP. However, when we illustrate a scenario with moderate synchronised loss, we see the throughputs of connection 1 and 2 diverge quite quickly. This poses a warning that HSTCP running in a droptail environment may show undesirable RTT unfairness. This curve is worse than the linear curve shown for synchronised losses in standard TCP shown in [1].

VI. Simulation Verification

We conduct simulation using NS-2 to verify that our model predicts well the throughput of two connections with varying RTT. Our simulation scripts are based on the HSTCP test suite supplied with NS-2.26. This is a simple dumbell topology as shown in Figure (3). We introduce variablity to one of the sources' links to the bottleneck router, and make the bottleneck capacity variable. We choose the scenario based on the "good" queue, which is specified to be an Adaptive RED queue of size 200, with an average queueing delay of 5ms.

We ran simulations with bandwidths of 1.5Mbps, 10Mbps and 100Mbps with fixed longer RTTs of 500ms, 500ms and 250ms respectively. Packet size is 576Bytes and maximum window size is set to 83000 to make sure the TCP window size is limited only by the network. Each simluation ran for 1500 seconds, the throughput calculated over the last 1000 seconds. For each ratio point, we repeat a simulation three times and plot the thoughput.

Figure (4) shows simulation results for a bottleneck bandwidth of 1.5Mbps, and fixed longer RTT of 500ms. Plotted

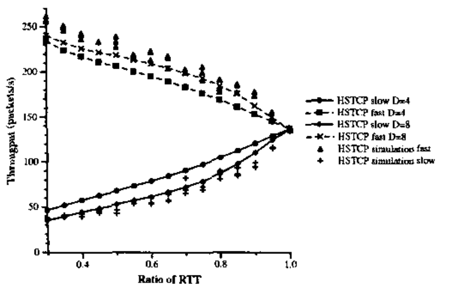

Fig. 4. Comparison of RTT fairness of HSTCP model and simulation. Bandwidth = 1.5Mbps, packet size = 576bytes, slow RTT = 0.5 seconds.

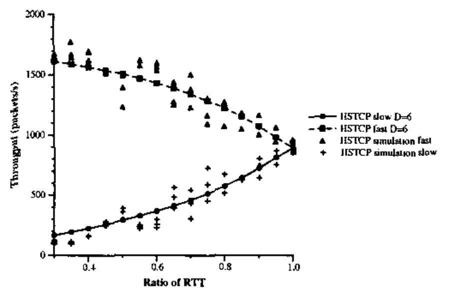

Fig. 5. Comparison of RTT fairness of HSTCP model and simulation. Bandwidth = 10Mbps, packet size = 576bytes, slow RTT = 0.5 seconds.

against the simulation results are two sets of results from our model with two different loss synchronisation levels, $D = 4$ and $D = 8$. The latter model suggests that this simulation scenario experienced a moderate level of loss synchronisation. Indeed from examination of the simulation trace, seven losses per congestion event were common. At a low number of packets per second, the Adaptive RED buffer is making a comparatively low number of updates to the average queue length per second. This allows the instantaneous queue length to grow well past the maximum threshold causing a high number of drops when the average queue length begins to catch up. Therefore, the two flows experience a moderate level of loss synchronisation, as at each congestion event there is a high probability of lost packets from both connections.

Figure (5) shows simulation results versus model results

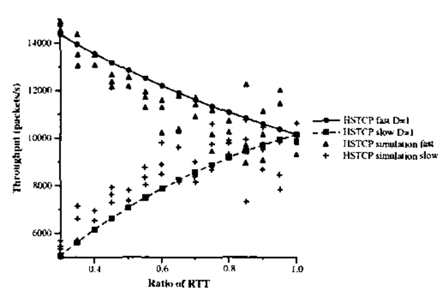

Fig. 6. Comparison of RTT fairness of HSTCP model and simulation. Bandwidth = 100Mbps, packet size = 576bytes, slow RTT = 0.25 seconds.

for a bandwidth of 10Mbps. Note the close relationship when $D = 6$, showing that less loss synchronisation occurs at a higher bandwidth. With more packets per second traversing the Adaptive RED queue, the average queue length tracks the instantaneous queue length more closely and can detect increase in queue length earlier.

Figure (6) shows a high congestion window scenario involving a shared bandwidth of 100Mbps. Note the convex shape of both the model and simulation results. At this high bandwidth, the Adaptive RED buffer is able to detect increases in queue length early enough to signal the respective signal before a second packet is dropped. This is due to the high number of updates to the average queue length state variable, which brings the buffer within the packet drop thresholds earlier.

VII. Conclusion

In this paper, analiytic models have been developed for Highspeed TCP congestion avoidance and fairness. The models can include both flows of synchronous and non-synchronous loss.

Our model shows that an Adaptive RED queueing mechanism can still cause synchronised losses for two competing long lived flows. We confirm that, much like standard TCP, a non-synchronised loss scenario provides better thoughput and improved RTT fairness. However, we show that, for a scenario of moderate loss synchronisation possible in Droptail buffers, Highspeed TCP has the potential to exhibit differences in throughput that which diverge quickly with respect to ratio of RTTs.

We have shown that an Adaptive RED buffer is better able to prevent cases of synchronous loss for higher bandwidth links. Therefore, it is recommended that some form of Active Queue Management is required to maintain similar RTT fairness in HSTCP as for standard TCP.

Extensive simulation has validated our results.

References

[1] E. Altman, C. Barakat, E. Laborde, P. Brown, and D. Collange, "Fairness analysis of TCP/IP," in *Proceedings of the IEEE Conference on Decision and Control (CDC'00)*, 2000.

[2] S. Floyd, "Highspeed TCP for large congestion windows," IETF, Tech. Rep. RFC3649, 2003. [Online]. Available: http://www.ietf.org/rfc/rfc3649.txt?number=3649

[3] C. Jin, D. Wei, and S. Low, "FAST TCP: Motivation, architecture, algorithms, and performance," in *INFOCOM*, 2004.

[4] T. Kelly, "Scalable tcp: improving performance in highspeed wide area networks," *SIGCOMM Comput. Commun. Rev.*, vol. 33, no. 2, pp. 83–91, 2003.

[5] T. V. Lakshman and U. Madhow, "The performance of TCP/IP for networks with high bandwidth-delay products and random loss," *IEEE/ACM Trans. Netw.*, vol. 5, no. 3, pp. 336–350, 1997.

[6] J. Padhye, V. Firoiu, D. F. Towsley, and J. F. Kurose, "Modeling tcp reno performance: a simple model and its empirical validation," *IEEE/ACM Transactions on Networking (TON)*, vol. 8, no. 2, pp. 133–145, 2000.

[7] B. Sikdar, S. Kalyanaraman, and K. S. Vastola, "Analytic models for the latency and steady-state throughput of TCP Tahoe, Reno, and SACK," *IEEE/ACM Trans. Netw.*, vol. 11, no. 6, pp. 959–971, 2003.

[8] S. Floyd, R. Gummadi, and S. Shenker, "Adaptive red: an algorithm for increasing the robustness of red's active queue management, http://www.icir.org/floyd/adaptivered/," 2001.