

# On RTT Scaling in H-TCP

D.J.Leith, R.N.Shorten

Hamilton Institute, National University of Ireland Maynooth  
email: {doug.leith, robert.shorten}@nuim.ie

## I. INTRODUCTION

This note discusses the impact of RTT scaling in the proposed H-TCP modifications to the additive increase element of the TCP AIMD congestion control algorithm [1]. To provide a baseline for comparison, we first detail some fundamental characteristics of both the standard TCP algorithm and the basic H-TCP algorithm without RTT scaling. The impact of RTT scaling is then discussed. Since the proposed H-TCP modifications are to the AIMD component of the TCP congestion control algorithm, leaving changes to slow start as a separate issue, our focus in this document is primarily on the behaviour of long-lived flows.

## II. STANDARD TCP

In outline, the standard TCP congestion control algorithm updates the congestion window  $cwnd$  according to an Additive Increase Multiplicative Decrease (AIMD) control law. In the congestion avoidance phase, when a source  $i$  receives a TCP ACK, it increments  $cwnd$  according to  $cwnd \rightarrow cwnd + \alpha^s/cwnd$  where  $\alpha^s = 1$  for the standard TCP algorithm. When packet loss is detected,  $cwnd$  is reduced by a backoff factor  $\beta^s$ : thus  $cwnd \rightarrow \beta^s cwnd$ , where  $\beta^s = 0.5$  for standard TCP.

Let  $w_i(k)$  denote the congestion window size of flow  $i$  immediately before the  $k$ 'th network congestion event is detected<sup>1</sup>. We have immediately from the AIMD algorithm that

$$w_i(k+1) = \beta_i(k)w_i(k) + \alpha_i T(k) \quad (1)$$

where  $\beta_i(k) = \beta^s$  if flow  $i$  sees a drop at the  $k$ th congestion event and backs off while  $\beta_i(k) = 1$  if flow  $i$  does not back off.  $T(k)$  is the duration (in seconds) of the  $k$ th congestion epoch and  $\alpha_i$  denote the effective increase rate of flow  $i$  in packets/s; that is, while  $\alpha^s$  determines the rate of increase per round-trip time,  $\alpha_i$  is the corresponding effective rate of increase per second. Observe that  $\alpha_i$  is therefore approximately  $1/RTT_i$ , where  $RTT_i$  is the round-trip time of flow  $i$ .

We note the following properties of the standard TCP AIMD algorithm for long-lived flows in a dumbbell topology

- *Fairness.* Flows with the same round-trip time have, on average, the same throughput and congestion window

<sup>1</sup>A network congestion event occurs when one or more flows experience packet loss. Note that later in this note we sometimes also use the quantity  $\Omega_i$  rather than  $w_i$ .  $\Omega_i$  and  $w_i$  are different quantities. Namely,  $\Omega_i$  is the peak congestion window of flow  $i$  immediately before that flow backs off – it is thus similar to the quantity used in Padhye's fluid model. Whereas  $w_i$  is the value of the congestion window of flow  $i$  conditioned on any flow in the network backing off. The latter is a useful quantity when analysing the dynamics of a network and interactions between flows. We have that  $w_i = \Omega_i$  in the case of synchronised drops

$E[w_i]$ . Unfairness generally exists between flows with different round-trip times. This is illustrated in Figure 2, which plots the ratio of the mean congestion windows  $E[w_i]$  as the path propagation delay of one flow is varied. Also shown in Figure 2 are the values predicted by

$$\frac{E[w_i]}{E[w_j]} = \frac{\alpha_i/(1 - E[\beta_i])}{\alpha_j/(1 - E[\beta_j])} \approx \frac{1/\lambda_i RTT_i}{1/\lambda_j RTT_j} \quad (2)$$

where  $\lambda_i$  is the probability of flow  $i$  experiencing a packet loss when a network congestion event occurs (so, for example,  $\lambda_i = 1$  when drops are synchronised). This formula is derived in [2], [3]. Also marked by solid lines in this figure is the ratio predicted when  $\lambda_i$  is the same for both flows. These lines are marked on most RTT unfairness figures in this note to provide a reference for comparing plots. Figure 3 shows the corresponding results obtained for 10 competing flows with a range of round-trip times as the RTT of flow 1 is varied (the distribution of flow round-trip times used is similar to that in [4]). Note that these examples include background web traffic with a range of connection lengths (the web traffic generator in *NS* is described in [5]).

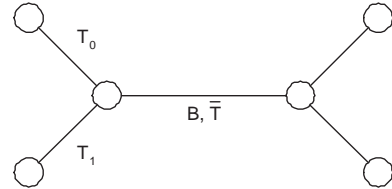


Fig. 1. Dumbbell topology.

- *Convergence Rate.* Following startup of a new flow or other disturbance, the network converges to equilibrium in approximately 4 congestion epochs. More precisely, the mean congestion windows  $w_i$  of the flows converge to within 95% of their equilibrium values in no more than  $\log(0.05)/\log(\max E[\beta_i])$  congestion epochs, which yields 4 epochs when  $E[\beta_i] = 0.5$  [6], [3]. This is illustrated in Figure 4 (in this example packets drops are synchronised but the unsynchronised drop situation is identical provided we work in terms of the mean peak congestion window).
- *Loss Overhead.* Suppose a TCP flow loses on average  $n_{lost}$  packets at a congestion event. For a single flow  $i$ , on average the number of packets,  $n_{total}$ , sent between congestion events is  $1/2(1 + \beta^s)(1 - \beta^s)\Omega_i^2/\alpha^s$ , where this expression is obtained by simply counting the number of packets under the TCP sawtooth and  $\Omega_i$  is the value of

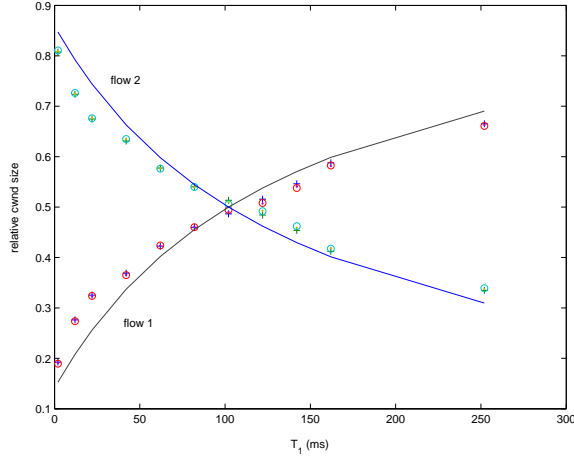


Fig. 2. Variation of time average  $w_i(k)$  with propagation delay  $T_1$  in dumbbell topology of Figure 1. Key: +NS simulation result;  $\circ$  equation (2); solid lines correspond to equal drop probability case. (Network parameters:  $B=100\text{Mb}$ ,  $q_{max}=80$  packets,  $\bar{T}=20\text{ms}$ ,  $T_0=102\text{ms}$ ; approximately 0.5% bidirectional background web traffic).

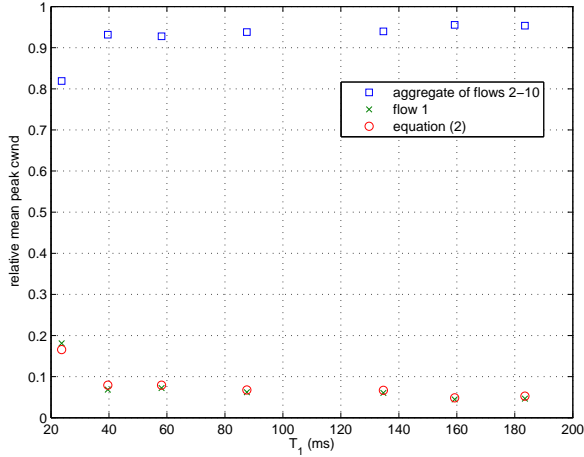


Fig. 3. Fairness between 10 competing TCP flows as RTT of flow 1 is varied. (NS simulation, dumbbell topology, 155Mb bottleneck link,  $\bar{T}$  10ms, 10 flows, flow 1 access link delay  $T_1$  marked on x-axis, flow 2-10 access link delays  $T_i$  0,0,2,8,20,44,80,138,200 ms, queue 250 packets, approx. 0.5% bidirectional web traffic).

flow  $i$ 's  $cwnd$  immediately before the flow backs off. The loss overhead (lost packets as a fraction of transmitted packets) is given by  $n_{lost}/n_{total}$ . Values for standard TCP are shown in Table I for  $n_{lost} = 1$ . Observe that the loss overhead is highest for small peak congestion windows, as might be expected.

- **Congestion Epoch Duration.** It is also useful to consider the variation of the congestion epoch duration with peak congestion window  $\Omega_i$ . We have that the congestion epoch duration for a flow, measured in round-trip times, is  $(1 - \beta^s)\Omega_i/\alpha^s$ . Thus, the duration in seconds is  $T = (1 - \beta^s)\Omega_i RTT_i/\alpha^s$ . Values for standard TCP are shown in Table II. It can be seen that the congestion epoch duration quickly becomes very long for large congestion windows, resulting in poor responsiveness of the TCP

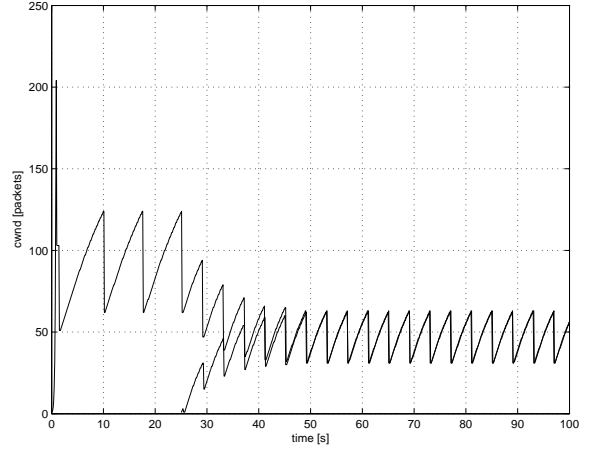


Fig. 4. NS packet-level simulation ( $\alpha_i = 1$ ,  $\beta_i = 0.5$ , dumb-bell with 10Mbps bottleneck bandwidth, 100ms propagation delay, 40 packet queue).

$\Omega_i$ (packets)	Loss overhead
10	$2.67 \times 10^{-2}$
100	$2.67 \times 10^{-4}$
1000	$2.67 \times 10^{-6}$
2000	$6.67 \times 10^{-7}$
5000	$1.06 \times 10^{-7}$
10000	$2.67 \times 10^{-8}$
20000	$6.67 \times 10^{-9}$
50000	$1.06 \times 10^{-9}$

TABLE I

LOSS OVERHEAD VS PEAK  $cwnd$ .

congestion control algorithm (recall that the convergence time above is stated in terms of congestion epochs and thus increases in proportion with the congestion epoch duration).

$\Omega_i$ (packets)	Congestion epoch duration (seconds)			
	RTT 10ms	RTT 50ms	RTT 100ms	RTT 250ms
10	0.05	0.25	0.5	1.25
100	0.5	2.5	5	12.5
1000	5	25	50	125
2000	10	50	100	250
5000	25	125	250	625
10000	50	250	500	1250
20000	100	500	1000	2500
50000	250	1250	2500	6250

TABLE II

CONGESTION EPOCH DURATION (IN SECONDS) VS PEAK CONGESTION WINDOW.

### III. H-TCP WITHOUT RTT SCALING

The basic H-TCP proposal[1] modifies the additive increase algorithm to

$$cwnd \leftarrow cwnd + \frac{f_{\alpha^s}(\Delta)}{cwnd} \quad (3)$$

with

$$f_{\alpha^s}(\Delta) = \begin{cases} 1 & \Delta \leq \Delta_L \\ \bar{f}_{\alpha^s}(\Delta) & \Delta \geq \Delta_L \end{cases} \quad (4)$$

where  $\Delta_L$  is a specified threshold such that the standard TCP update algorithm is used while  $\Delta \leq \Delta_L$ . A quadratic increase

function  $\bar{f}_{\alpha^s}$  is suggested in [1], [7], namely

$$\bar{f}_{\alpha^s}(\Delta) = 1 + 10(\Delta - \Delta_L) + 0.25(\Delta - \Delta_L)^2 \quad (5)$$

H-TCP has similar fairness and convergence properties to the standard TCP algorithm.

- *Fairness.* Figures 5 and 6 illustrate the RTT unfairness characteristics of competing H-TCP flows. It can be seen that the behaviour is very similar to that of standard TCP.

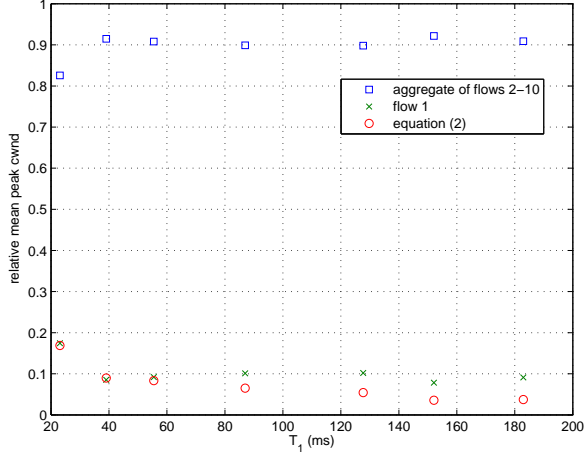


Fig. 6. Fairness between 10 competing H-TCP flows as RTT of flow 1 is varied. (NS simulation, dumbbell topology, 500Mb bottleneck link,  $\bar{T}$  10ms, 10 flows, flow 1 access link delay marked on x-axis, flow 2-10 access link delays 0,0,2,8,20,44,80,138,200 ms, queue 250 packets, approx. 0.5% bidirectional web traffic).

- *Convergence Rate.* The convergence rate of competing H-TCP flows is illustrated in Figure 7. As with standard TCP, convergence following a network disturbance takes approximately 4 congestion epochs.

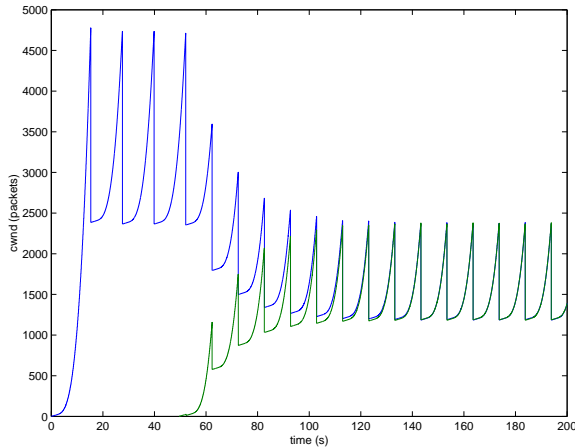


Fig. 7. Example of two H-TCP flows illustrating rapid convergence to fairness - taking approximately 4 congestion epochs which is in agreement with the rise-time analysis for  $\beta_i = 0.5$  (NS simulation, network parameters: 500Mb bottleneck link, 100ms delay, queue 500 packets).

The more aggressive increase function  $f_{\alpha^s}$  used in H-TCP decreases the congestion epoch duration for large congestion windows while increasing the loss overhead. In steady

state we have for flow  $i$  that,

$$(1 - \beta^s)\Omega_i = 1/RTT_i \int_0^T f_{\alpha^s}(\Delta)d\Delta \quad (6)$$

where  $T$  is the congestion epoch duration. Evaluating this for the increase function in (5) yields

$$(1 - \beta^s)\Omega_i = 1/RTT_i[T + 5(T - 1)^2 + (T - 1)^3/12] \quad (7)$$

Numerically solving this nonlinear equation for the congestion epoch duration,  $T$ , we obtain the values given in Table III. The impact of the more aggressive increase function in reducing the congestion epoch duration with large congestion windows is evident.

$\Omega_i$ (packets)	Congestion epoch duration (seconds)			
	RTT 10ms	RTT 50ms	RTT 100ms	RTT 250ms
10	0.05	0.25	0.5	1.1
100	0.5	1.4	1.8	2.4
1000	1.8	3.05	4.0	5.7
2000	2.2	4.0	5.2	7.6
5000	3.0	5.7	7.6	11.2
10000	4.0	7.6	10.2	15.1
20000	5.2	10.2	13.7	19.3
50000	7.6	15.1	20.3	29.9

TABLE III

CONGESTION EPOCH DURATION (IN SECONDS) WITH BASIC H-TCP SCHEME VS PEAK CONGESTION WINDOW.

We can obtain the associated loss overhead by observing that the number of packets  $n_{total}$  sent between congestion event by flow  $i$  is

$$n_{total} = \int_0^T cwnd_i(t)/RTT_i dt \quad (8)$$

That is, for the basic H-TCP scheme

$$n_{total} = \{\beta^s\Omega_i T + \int_0^T f_{\alpha^s}(\Delta)/RTT_i d\Delta\}/RTT_i \quad (9)$$

Evaluating this for the increase function in (5) yields

$$n_{total} = \beta^s\Omega_i T/RTT_i + \{1/2T^2 + 5/3(T-1)^3 + 1/48(T-1)^4\}/RTT_i^2 \quad (10)$$

Using the values from Table III for the congestion epoch duration  $T$ , we then obtain the loss overhead values  $1/n_{total}$  shown in Table IV. Observe that the highest loss overhead is associated with small peak congestion windows and the worst case is thus identical to that for standard TCP.

We can assess the friendliness of H-TCP flows when competing with legacy TCP flows by comparing the effective increase rate of an H-TCP flow with that of a standard TCP flow. The effective increase rate in packets per round-trip time is given by  $(\Omega_i - \beta^s\Omega_i)/(T/RTT_i)$ . For standard TCP this is approximately 1. Table III details the relationship between the congestion epoch duration and the peak congestion window  $\Omega_i$ . Using this relationship, we obtain the effective increase rates in Table V for H-TCP. The unfairness that occurs when H-TCP flows compete with legacy TCP flows with the same round-trip time is roughly proportional to the difference in increase rates. Figure 11 also shows simulation results illustrating the dependence of fairness between H-TCP and

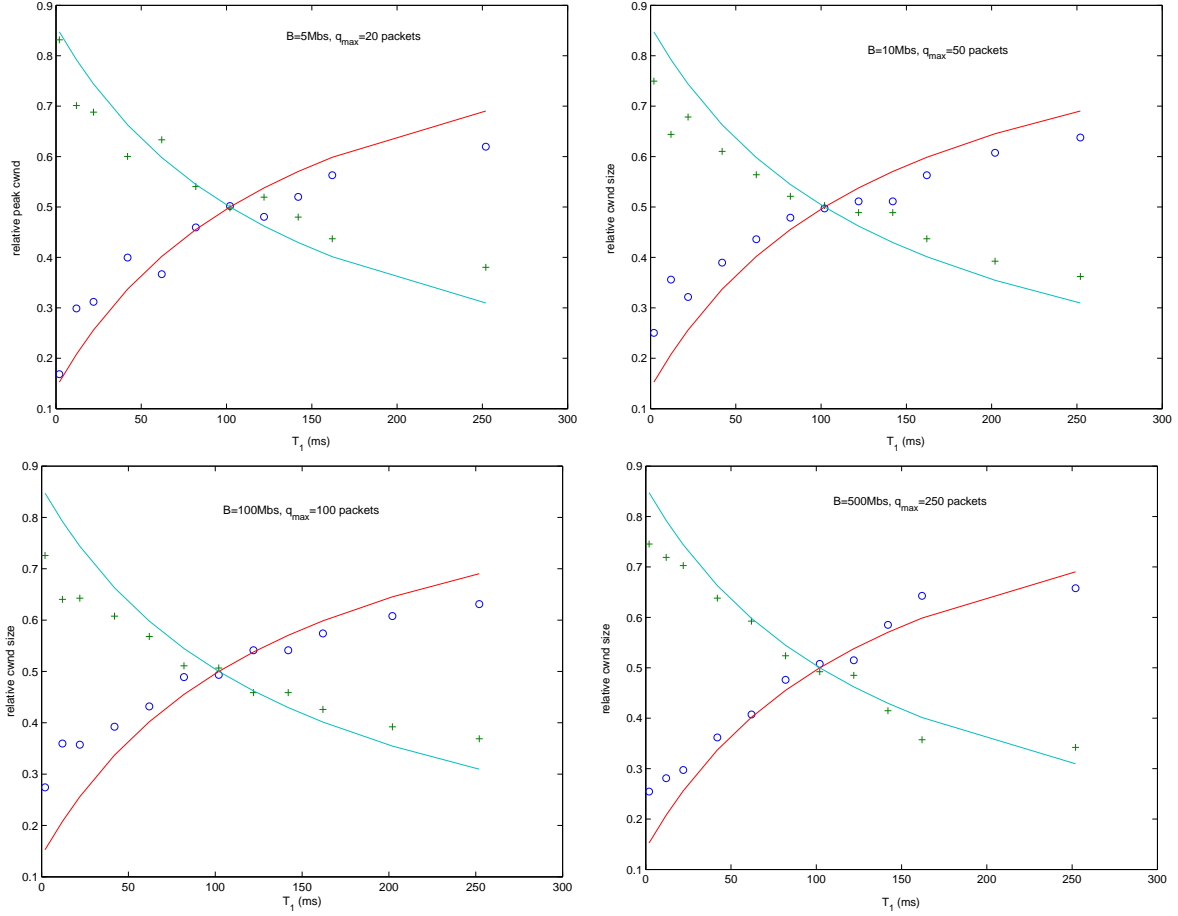


Fig. 5. Fairness between competing H-TCP flows as RTT of second flow is varied. Key: + flow 1, o flow 2, solid lines correspond to equal drop probability case for standard TCP. (*NS* simulation, dumbbell topology, common network parameters:  $\bar{T}=20\text{ms}$ ,  $T_0=102\text{ms}$ ; approximately 0.5% bidirectional background web traffic).

$\Omega_i$ (packets)	Loss overhead			
	RTT 10ms	RTT 50ms	RTT 100ms	RTT 250ms
10	$2.67 \times 10^{-2}$	$2.67 \times 10^{-2}$	$2.67 \times 10^{-2}$	$3.0 \times 10^{-2}$
100	$2.67 \times 10^{-4}$	$5.1 \times 10^{-4}$	$8.7 \times 10^{-4}$	$1.6 \times 10^{-3}$
1000	$8.7 \times 10^{-6}$	$2.6 \times 10^{-5}$	$4.0 \times 10^{-5}$	$6.8 \times 10^{-5}$
2000	$3.6 \times 10^{-6}$	$9.9 \times 10^{-6}$	$1.5 \times 10^{-5}$	$2.5 \times 10^{-5}$
5000	$1.0 \times 10^{-6}$	$2.7 \times 10^{-6}$	$4.1 \times 10^{-6}$	$6.7 \times 10^{-6}$
10000	$4.0 \times 10^{-7}$	$1.0 \times 10^{-6}$	$1.5 \times 10^{-6}$	$2.5 \times 10^{-6}$
20000	$1.5 \times 10^{-7}$	$3.8 \times 10^{-7}$	$5.6 \times 10^{-7}$	$9.4 \times 10^{-7}$
50000	$4.0 \times 10^{-8}$	$1.0 \times 10^{-7}$	$1.5 \times 10^{-7}$	$2.6 \times 10^{-7}$

TABLE IV

LOSS OVERHEAD VS PEAK CONGESTION WINDOW FOR BASIC H-TCP SCHEME.

$\Omega_i$ (packets)	Effective number of standard TCP flows			
	RTT 10ms	RTT 50ms	RTT 100ms	RTT 250ms
10	1.0	1.0	1	1.1
100	1.0	1.8	2.8	5.2
1000	2.8	8.2	12.5	21.9
2000	4.5	12.5	19.2	32.9
5000	8.3	21.9	32.9	55.8
10000	12.5	32.9	49.0	82.8
20000	19.2	49.0	73.0	129.5
50000	32.9	82.8	123.1	209.0

TABLE V

EFFECTIVE INCREASE RATE WITH BASIC H-TCP SCHEME VS PEAK CONGESTION WINDOW.

legacy TCP flows on the bandwidth-delay product (it can be seen that the figures are in good agreement with Table V; results are not shown for bandwidth-delay product above 1000 because the lengthening congestion epoch duration of standard TCP means that extremely long time histories are required in order to collect good statistics).

#### Comment: Smooth transition to high-speed operation.

The increase function (5) is a continuous function of the time since last backoff,  $\Delta$  and so as  $\Delta$  increases there is no abrupt

change in the increase rate. This ensures a smooth transition from low to high-speed operation under network conditions, see for example Figure 8 which shows time histories of a H-TCP flow competing with a legacy TCP flow in the transition regime where the H-TCP flow just enters high-speed mode towards the end of each congestion epoch.

#### IV. H-TCP WITH RTT SCALING

In this section we consider the impact of amending the basic H-TCP scheme to include RTT scaling; that is, changing the

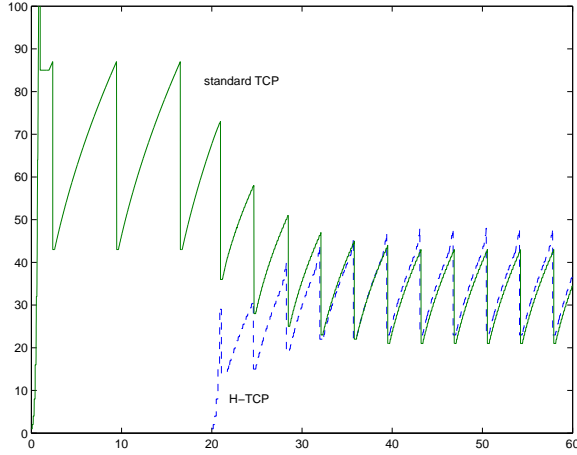


Fig. 8. Example of an H-TCP flow and a legacy TCP in the transition regime where the H-TCP flow is just entering high-speed mode. (NS simulation, network parameters: 5Mb bottleneck link, 100ms delay, queue 44 packets).

congestion window increase function to be

$$f_{\alpha^s}(\Delta) = \begin{cases} 1 & \Delta \leq \Delta^L \\ \max[1, \rho \bar{f}_{\alpha^s}(\Delta)] & \Delta \geq \Delta^L \end{cases} \quad (11)$$

where the scaling factor  $\rho$  is given by the round-trip time normalised by a reference round-trip time  $\overline{RTT}$ ; that is, for flow  $i$   $\rho = RTT_i / \overline{RTT}$ . We return later to the choice of  $\overline{RTT}$ . To guarantee backward compatibility with standard TCP in low-speed regimes, the increase rate  $f_{\alpha^s}$  is constrained to be greater than or equal to one. Furthermore, it is prudent to restrict the value of the scaling factor  $\rho$ . For illustrative purposes, we constrain  $\rho$  to lie in the interval  $[0.1, 2]$ ; for  $\overline{RTT} = 100ms$ , this corresponds to limiting RTT scaling to round-trip times in the range  $[10ms, 200ms]$ .

The impact on fairness of this RTT scaling modification is illustrated in Figures 9 and 10. It can be seen that, as expected, RTT unfairness remains largely unchanged in low-speed networks (the H-TCP algorithm behaves as standard TCP in low-speed conditions and hence the RTT unfairness exhibited by the standard TCP algorithm continues to be present) but RTT unfairness is greatly reduced in higher-speed conditions.

We argue that mitigating RTT unfairness is not, however, necessarily the primary benefit of RTT scaling in high-speed networks. We observe that RTT scaling yields the following additional benefits,

- *Improved Friendliness.* By reducing the aggressiveness of the H-TCP algorithm on short RTT paths, RTT scaling improves the friendliness of H-TCP flows when competing against standard TCP. This is illustrated, for example, in Figure 11. In more detail, with RTT scaling and  $\overline{RTT} = 100ms$  we have that the congestion epoch duration roughly corresponds to the 100ms column in Table III. It is necessary, however, to take account of the impact of constraining the RTT scaling factor  $\rho$  to the interval  $[0.1, 2]$  and the effect of constraining the increase rate to be at least 1 packet per round-trip time for backward compatibility. The congestion epoch duration,

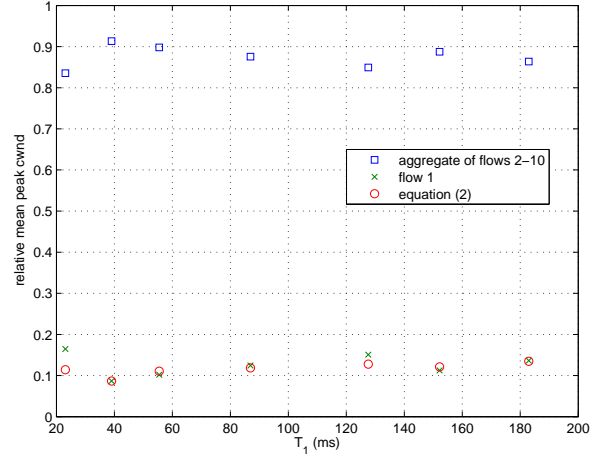


Fig. 10. Fairness between 10 competing H-TCP flows as RTT of flow 1 is varied. (NS simulation, dumbbell topology, 500Mb bottleneck link,  $\overline{T}$  10ms, 10 flows, flow 1 access link delay marked on x-axis, flow 2-10 delays 0,0,2,8,20,44,80,138,200 ms, queue 250 packets, approx. 0.5% bidirectional web traffic).

adjusted for these factors, is shown in Table VI. The corresponding effective increase rates are shown in Table VII.

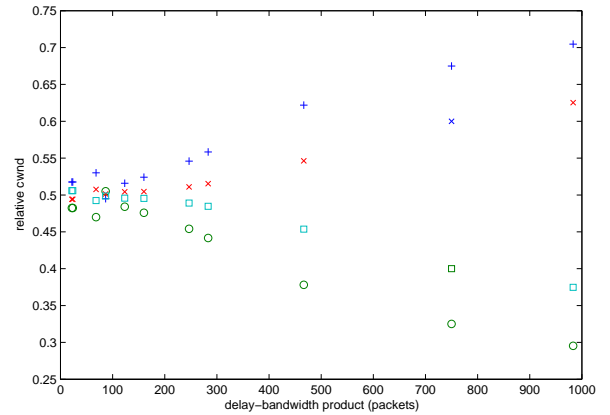


Fig. 11. Fairness (friendliness) between an H-TCP flow with/without RTT scaling competing against a standard TCP flow as delay-bandwidth product is varied. Key: + H-TCP without RTT scaling, o competing standard TCP, x H-TCP with RTT scaling  $\overline{RTT}=100ms$ , □ competing standard TCP flow. (NS simulation, network parameters:  $T=20ms$ ,  $T_0=2ms$ ,  $T_1=2ms$ ,  $B \in \{1, 2, 10, 20, 40, 60, 80, 100, 200, 400\}Mb$ ,  $q_{max}=20$  packets for  $B < 10Mb$ , 50 packets for  $B < 80Mb$ , 100 packets for  $B < 400Mb$ , 250 packets for  $B=400Mb$ ; approximately 0.5% bidirectional background web traffic).

- *Reduced Loss Overhead.* Similarly, RTT scaling reduces the H-TCP loss overhead on paths with short round-trip times: see Table VIII.

These benefits are obtained at the price of a longer congestion epoch duration on short round-trip time paths when RTT scaling is used. As discussed earlier, network responsiveness becomes slower as the congestion epoch duration becomes longer. However, on short round-trip time paths the congestion epoch duration is very short when RTT scaling is not used.

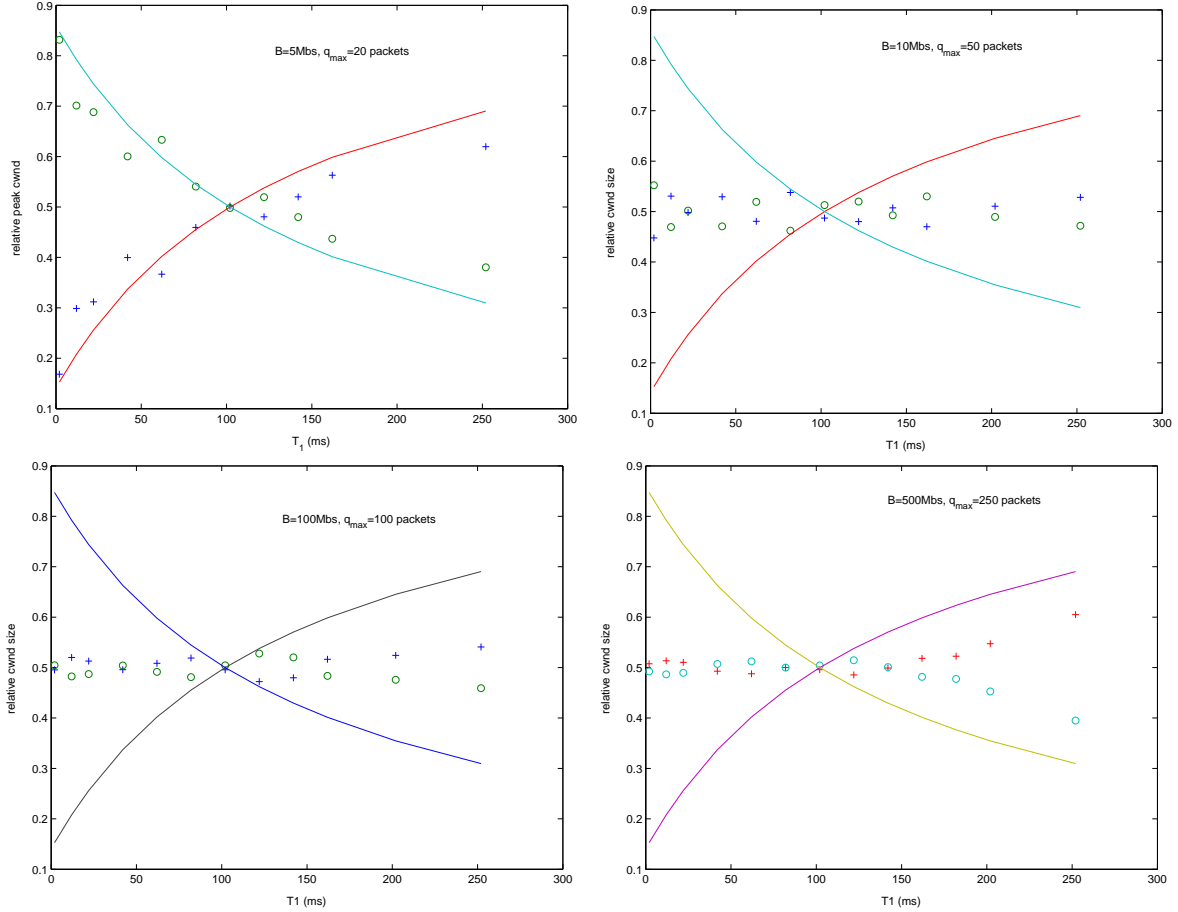


Fig. 9. Fairness between competing H-TCP flows with RTT scaling as RTT of second flow is varied. Key: + flow 1, o flow 2, solid lines correspond to equal drop probability case. (*NS* simulation, dumbbell topology, common network parameters:  $\bar{T}=20\text{ms}$ ,  $T_0=102\text{ms}$ ; approximately 0.5% bidirectional background web traffic;  $\overline{RTT}=100\text{ms}$ ).

$\Omega_i$ (packets)	Congestion epoch duration (seconds)			
	RTT 10ms	RTT 50ms	RTT 100ms	RTT 250ms
10	0.05	0.25	0.5	1.1
100	0.5	1.7	1.8	1.8
1000	3.5	3.9	4.0	4.0
2000	4.9	5.2	5.2	5.2
5000	7.4	7.6	7.6	7.6
10000	10.1	10.2	10.2	10.2
20000	13.7	13.7	13.7	13.7
50000	20.3	20.3	20.3	20.3

TABLE VI

CONGESTION EPOCH DURATION (IN SECONDS) WITH H-TCP RTT SCHEDULING SCHEME VS PEAK CONGESTION WINDOW ( $\overline{RTT}=100\text{ms}$ ).

$\Omega_i$ (packets)	Effective number of standard TCP flows			
	RTT 10ms	RTT 50ms	RTT 100ms	RTT 250ms
10	1.0	1.0	1	1.1
100	1.0	1.5	2.8	6.9
1000	1.4	6.4	12.5	31.2
2000	2.0	9.6	19.2	47.2
5000	3.4	16.4	32.9	82.2
10000	4.9	24.5	49.0	122.5
20000	7.3	36.5	73.0	182.5
50000	12.3	61.6	123.1	307.8

TABLE VII

EFFECTIVE INCREASE RATE WITH H-TCP AND RTT SCALING VS PEAK CONGESTION WINDOW ( $\overline{RTT}=100\text{ms}$ ).

An increase in congestion epoch duration can thus potentially be tolerated without unduly reducing network responsiveness from a user viewpoint.

The responsiveness is determined by the value of the reference value  $\overline{RTT}$ . In more detail, we have that H-TCP exhibits the same rise time, measured in congestion epochs, as standard TCP; that is, a 95% rise time of 4 congestion epochs when  $E[\beta_i] = 0.5$ . We can select  $\overline{RTT}$  to ensure a specified bound on the congestion epoch duration and hence meet a requirement on the actual (measured in seconds rather than congestion epochs) rise time. The congestion epoch duration

versus peak congestion window and round-trip time is shown in Table III for the basic H-TCP algorithm. With RTT scaling, the dependence of the congestion epoch duration on round-trip time is effectively removed and the value of  $\overline{RTT}$  then selects the particular column in Table III that applies - in the results presented in this note a value of 100ms is used for  $\overline{RTT}$ . With this choice, the congestion epoch duration is less than 10 seconds when the congestion window size is less than 10,000 packets; that is, with  $\overline{RTT} = 100\text{ms}$  the 95% rise time is less than 40 s for all congestion window sizes less than 10,000 packets.

$\Omega_i$ (packets)	Loss overhead			
	RTT 10ms	RTT 50ms	RTT 100ms	RTT 250ms
10	$2.67 \times 10^{-2}$	$2.67 \times 10^{-2}$	$2.67 \times 10^{-2}$	$3.0 \times 10^{-2}$
100	$2.67 \times 10^{-4}$	$4.6 \times 10^{-4}$	$8.7 \times 10^{-4}$	$2.1 \times 10^{-3}$
1000	$4.4 \times 10^{-6}$	$2.0 \times 10^{-5}$	$4.0 \times 10^{-5}$	$9.9 \times 10^{-5}$
2000	$1.6 \times 10^{-6}$	$7.5 \times 10^{-6}$	$1.5 \times 10^{-5}$	$3.7 \times 10^{-5}$
5000	$4.2 \times 10^{-7}$	$2.0 \times 10^{-6}$	$4.1 \times 10^{-6}$	$1.0 \times 10^{-5}$
10000	$1.5 \times 10^{-7}$	$7.6 \times 10^{-7}$	$1.5 \times 10^{-6}$	$3.8 \times 10^{-6}$
20000	$5.6 \times 10^{-8}$	$2.8 \times 10^{-7}$	$5.6 \times 10^{-7}$	$1.4 \times 10^{-6}$
50000	$1.5 \times 10^{-8}$	$7.5 \times 10^{-8}$	$1.5 \times 10^{-7}$	$3.9 \times 10^{-7}$

TABLE VIII

LOSS OVERHEAD VS PEAK CONGESTION WINDOW FOR H-TCP AND RTT SCALING ( $\overline{RTT}=100\text{ms}$ ).

## V. RTT SCALING & FUTURE DEVELOPMENT

When considering modifications to TCP, it is prudent to try to assess the impact that proposed changes may have in terms of opening or closing off avenues of future development. In this context, we recall that previous work [8] on the application of RTT scaling in low speed environments concluded that it is difficult to achieve backward compatibility. In particular, it is difficult to define a reference value  $\overline{RTT}$  that avoids either the scaled flows gaining an unfair bandwidth share when competing against standard TCP, or vice versa. This result is not surprising - for round-trip times shorter than  $\overline{RTT}$  RTT scaling reduces the AIMD increase parameter to a value below that of the standard TCP algorithm, while for round-trip times longer than  $\overline{RTT}$  RTT scaling increases the AIMD increase parameter above that of the standard TCP algorithm. Hence, fair co-existence seems impossible and the AIMD approach adopted in the standard TCP algorithm, with its associated RTT unfairness, therefore seems to impose some limit on the possible options going forward. In the context of high-speed networks, the requirement for friendliness towards standard TCP flows is relaxed and it is this that allows us to use RTT scaling during high-speed operation (RTT unfairness during low-speed operation remains, of course, similar to that for standard TCP in order to ensure backward compatibility).

With regard to the future, it therefore appears that roll-out of an AIMD-based high-speed protocol that continues to behave unfairly would seem to create a backward compatibility requirement in the future that would make it difficult for subsequent modifications to improve fairness. Conversely, it may be that an opportunity exists to improve the fairness between competing TCP flows by using, for example, RTT scaling. The backward compatibility requirement on subsequent developments of TCP would then have the virtue of maintaining fairness.

## REFERENCES

- [1] D.J.Leith and R.N.Shorten, "H-TCP: TCP congestion control for high bandwidth-delay product paths," *IETF Internet Draft draft-leith-tcp-htcp-00.txt*, June, 2005.
- [2] R.N.Shorten, F.Wirth, and D.J.Leith, "A positive systems model of TCP-like congestion control: Asymptotic results," *IEEE Transactions on Networking*, in press, 2005.
- [3] R.N.Shorten, D.J.Leith, and F.Wirth, "Modelling TCP congestion control dynamics in drop-tail environments," *Automatica*, to appear, 2005.
- [4] E. S.Floyd, "Internet research needs better models," *HotNets*, 2002.

- [5] W. Willinger, M. S. Taqqu, R. Sherman, and D. V. Wilson, "Self-similarity through high-variability: statistical analysis of Ethernet LAN traffic at the source level," *IEEE/ACM Transactions on Networking*, vol. 5, no. 1, pp. 71–86, 1997.
- [6] R.N.Shorten, D.J.Leith, J.Foy, and R.Kilduff, "Analysis and design of congestion control in synchronised communication networks," *Automatica*, vol. 41, pp. 725–730, 2005.
- [7] R. Shorten and D. Leith, "H-tcp protocol for high-speed long-distance networks," in *Proc. 2nd Workshop on Protocols for Fast Long Distance Networks*. Argonne, Canada, 2004.
- [8] T. Henderson and R. Katz, "On improving the fairness of tcp congestion avoidance," in *Proceedings of IEEE GLOBECOM*, 1998.