2001

A Round Trip Time and Timeout Aware Traffic Conditioner for Differentiated Services Networks

Ahsan Habib

Bharat Bhargava
Purdue University, bb@cs.purdue.edu

Sonia Fahmy
Purdue University, fahmy@cs.purdue.edu

Report Number:
01-021

A ROUND TRIP TIME AND TIMEOUT AWARE TRAFFIC CONDITIONER FOR DIFFERENTIATED SERVICES NETWORKS

Ahsan Habib
Bharat Bhargava
Sonia Fahmy

Department of Computer Science
Purdue University
West Lafayette, IN 47907

CSD TR #01-021
November 2001

A Round Trip Time and Timeout aware Traffic Conditioner for Differentiated Services Networks

Ahsan Habib, Bharat Bhargava, Sonia Fahmy

Center for Education and Research in Information Assurance and Security (CERIAS),

and Department of Computer Sciences

Purdue University, West Lafayette, IN 47907-1398, USA

Abstract

TCP connection throughput is inversely proportional to the connection Round Trip Time (RTT). To mitigate TCP bias to short RTT connections, a differentiated services traffic conditioner can ensure connections with long RTTs do not starve when connections with short RTTs get all extra resources after achieving the target rates. Current proposals for RTT-aware conditioners work well for a small number of connections when most TCP connections are in the congestion avoidance phase. If there is a large number of TCP connections, however, connections time-out and go to slow start. We show that current RTT-aware conditioners over-protect long RTT flows and starve short RTT flows in this case. We design and evaluate a conditioner based on RTT as well as the Retransmission Time-out (RTO). The proposed RTT-RTO aware traffic conditioner works well for realistic situations with a large number of connections. Simulation results in a variety of situations confirm that the conditioner mitigates RTT bias.

Keywords: Traffic Conditioner, RTT, RTO, Quality of Service, Differentiated Services, Assured Forwarding.

I. Introduction

The differentiated services (diff-serv) architecture [1] is a scalable approach for Quality of Service (QoS) in IP networks. The diff-serv model uses traffic conditioners at the edges of an administrative

This research is sponsored in part by the National Science Foundation grants CCR-001712 and CCR-001788, CERIAS, an IBM SUR grant, the Purdue Research Foundation, and the Schlumberger Foundation technical merit award.

domain to shape, mark, and drop traffic if necessary. The conditioner operations are based on bi-lateral Service Level Agreements (SLAs) between adjacent domains. In the core of the network, Per Hop Behaviors (PHBs) achieve service differentiation. Assured forwarding (AF) PHBs at core routers use an active queue management technique such as Random Early Detection [2] for IN and OUT of profile (RIO) packets [3].

TCP connection throughput is inversely proportional to the connection Round Trip Time (RTT). Traffic conditioners that mitigate this unfairness by being RTT-aware were first proposed in [4]: they avoid RTT bias of TCP connections through marking packets with high drop priority inversely proportional to the square of their RTTs according to the steady state TCP behavior. Such conditioners work well when the number of flows is small. We show in this paper that, for a large number of flows, *short RTT* flows time out in this case because only long RTT flows are protected by the conditioner after satisfying the target rate. Excess bandwidth is mostly given to long RTT flows. To remedy this unfairness introduced by an RTT-aware conditioner, we propose two strategies. The first strategy is to combine the RTT-aware conditioner with techniques that protect a TCP flow when its congestion window is small. We implement the small window protection with the RTT-aware traffic conditioner and show that small window protection removes the unfairness of the RTT-aware conditioner. The second method is to re-design the RTT-aware conditioner to consider time-outs as well as RTT to approximate throughput. We refer to this method as the RTT-RTO conditioner. The performance of the conditioner is analyzed both for data intensive applications and delay sensitive applications with realistic traffic models.

We note that our method for incorporating RTT-awareness into the conditioner does not grant all available resources to long-RTT connections while short-RTT connections starve. The RTT-awareness only mitigates unfairness in distributing excess bandwidth. When a network is under-provisioned, the

RTT-aware conditioner does not consider RTTs.

The rest of this paper is organized as follows. Section II discusses the basics of a conditioner. Section III summarizes previous work on diff-serv assured forwarding and intelligent traffic conditioners. Section IV explains our proposal for overcoming unfairness problems in RTT-aware traffic conditioners. Section V contains all the details of our simulation setup. Section VI presents the simulation results. We conclude with a summary and discussion of future work.

II. Basics of a Conditioner

A traffic conditioner may contain meters, markers, droppers, and shapers [1]. The conditioner may alter the temporal characteristics of a traffic stream to bring it into compliance with a traffic profile specified by the network administrator. Incoming traffic first passes through a classifier, which is used to select a class for each traffic flow. Then, the meter measures and sorts the classified packets into precedence levels. The decision (marking, shaping, or dropping) is based on the measurement results. Assured forwarding [5] provides up to three drop precedences (say DP0, DP1, and DP2) for each queue. The Differentiated Services Code Point (DSCP), contained in the IP header, is set to mark the precedence. When congestion occurs, packets marked with higher precedence must be dropped first.

Within each core assured service queue, discrimination among packets is done using a differential drop algorithm. The RIO algorithm distinguishes between two types of packets, IN and OUT of profile, using two RED instances. Each RED instance is configured using different parameters to achieve service differentiation. The Assured Forwarding PHB provides four classes (queues) of delivery for IP packets and three levels of drop precedence per class.

III. Related Work

In an early differentiated services paper, Clark and Fang show that sources with different target rates can approximately achieve their targets using RIO even for different Round Trip Times (RTTs), whereas

simple RED routers cannot [3]. With RIO, if two flows have same target rate and different RTTs, short RTT flows get most of the extra resources. Our goal is to distribute the extra resources among all flows such that short RTT flows do not steal all the extra bandwidth.

Ibanez and Nichols showed that target rates and TCP/UDP interaction are key factors in determining throughput of flows [6]. Seddigh, Nandy and Pieda showed that target rates and TCP/UDP interaction are also critical for the distribution of excess bandwidth in an over-provisioned network [7]. Fang, Seddigh and Nandy proposed the Time Sliding Window Three Color Marker (TSW3CM) [8], which we refer to as the standard conditioner throughout this paper.

Nandy et al extend the TSW marker to design RTT-aware traffic conditioners [4]. The basic idea of this conditioner is to adjust the packet drop rate in relation to the RTT. Hence, the acquired bandwidth for the aggregate becomes less sensitive to RTT. Their conditioner is based on the steady state TCP behavior as reported by Matthis et al [9], i.e., bandwidth is inversely proportional to RTT. Their model does not consider timeouts. However, we observe time-out events when a large number of flows is multiplexed onto a bottleneck. We discuss this further in the next section.

Feroz et al propose a TCP-Friendly marker [10]. As TCP applications over diff-serv are influenced by bursty packet loss behavior, they use TCP characteristics to design their marker. Their conditioner protects small-window flows from packet losses by marking such traffic as IN. The authors maintain spacing between IN and OUT tokens allocated to a flow to handle burstiness. Detailed analysis on a good window size threshold (below which a flow is marked as IN) for various situations is provided in [11]. We incorporate the idea of protecting small window flows into one of our RTT-aware traffic conditioner proposals.

IV. RTT-RTO Aware Conditioner

In this section, we discuss the design of a fair RTT-aware traffic conditioner. The RTT-aware traffic conditioner proposed in [4] avoids the TCP short RTT bias through marking packets with high drop priority inversely proportional to the square of their RTTs. This is based upon the steady state TCP behavior modeled in [9]. Equation (1) shows that, in this model, bandwidth is inversely proportional to RTT where MSS is the maximum segment size and p is the packet loss probability:

$$BW \propto \frac{MSS}{RTT\sqrt{p}} \tag{1}$$

The RTT-aware marking algorithm proposed in [4] works well when the number of flows is small because equation (1) accurately represents the fast retransmit and recovery behavior when p is small. We have observed that for a large number of flows, short RTT flows time out because only long RTT flows are protected by the conditioner after satisfying the target rates. Excess bandwidth is mostly given to long RTT flows.

To remedy this situation, we can use one of two strategies. First, we can avoid the problem by combining the RTT-aware conditioner with a technique that protects the TCP packets after time-outs. Feroz et al. propose the small window protection technique [10], which marks TCP packets with lowest drop priority when the congestion window of TCP is small. TCP grows the congestion window exponentially until it reaches the slow start threshold, *ssthresh*. The congestion window reduces to 1 or half of the *ssthresh* for timeouts or packet loss, respectively. Giving low drop priority to flows with small congestion window sizes helps these flows to achieve high throughput. The window size of a TCP connection is calculated at the conditioner using the sequence number of packets in the forward direction and the sequence number of acknowledgments (ACKs) in the opposite direction.

Analysis and performance of Small Window (SW) based conditioning is given in [10], [11]. With SW, a packet is marked as $DP0$ when the congestion window size of a particular flow is $< k$. In

this paper, we show that SW protects short RTT flows when an RTT-aware conditioner is used. This combination eliminates the unfairness of the basic RTT-aware conditioner for a large number of flows. The RTT-aware marking algorithm with SW is referred to as RTT-SW throughout the remainder of this paper.

The second approach to eliminate unfairness is to use the throughput approximation by Padhye et al [12], which considers timeouts. Equation (2) shows this approximation, where b is the number of packets acknowledged by a received ACK, and To is the timeout length:

$$BW \approx \frac{1}{RTT\sqrt{\frac{2bp}{3}} + To \times min(1, 3\sqrt{\frac{3bp}{8}})p(1+32p^2)} \tag{2}$$

If we take $b = \frac{3}{2}$ (one delayed ACK for two packets for every three incoming packets), approximate $min(1, 3\sqrt{\frac{3bp}{8}})$ to 1 (so that BW will be less than or equal to the right side of the equation (3)), and discard the higher order term of p, i.e., $32p^3$ (if p is small, p^3 will be very small), we can simplify equation (2) to:

$$BW \approx \frac{1}{RTT \times \sqrt{p} + To \times p} \tag{3}$$

Designing an RTT-aware traffic conditioner using equation (3) is more accurate than using equation (1). Consider two flows with achieved bandwidths BW_1 and BW_2. The objective is to obtain:

$$BW_1 = BW_2 \tag{4}$$

(3) and (4) give:

$$RTT_1 \times \sqrt{p_1} + To_1 \times p_1 = RTT_2 \times \sqrt{p_2} + To_2 \times p_2 \tag{5}$$

Let $\rho = \frac{p_2}{p_1}$. Equation (5) can then be written as:

$$RTT_1 \times \sqrt{p_1} + To_1 \times p_1 = RTT_2 \times \sqrt{\rho \times p_1} + To_2 \times \rho \times p_1 \tag{6}$$

Solving for ρ, this means that we should have:

$$\rho \propto \left(\frac{RTT_1}{RTT_2}\right)^2 \tag{7}$$

And:

$$\rho \propto \frac{To_1}{To_2} \tag{8}$$

Equations (7) and (8) show that the packet drop ratio between two flows depends on the square of ratio of RTT of the two flows and the ratio of their time-outs. We combine the two equations to obtain the following heuristic:

$$\rho^2 = \left(\frac{RTT_1}{RTT_2}\right)^2 \times \frac{To_1}{To_2} \tag{9}$$

We follow the same steps as in [4] to derive the marking probabilities. If measured rate is beyond the target rate of a flow, it marks the packet as $DP1$ or $DP2$ with probability $\frac{measuredRate - targetRate}{measuredRate}$. The ratio of $DP1$ and $DP2$ marked packets is directly related to the packet drop probabilities at the core. This means that packet drop at the core is proportional to the out-of-profile marked packets. Thus, equation (9) is used to marked packets as $DP1$ and $DP2$. The resulting algorithm, which we refer to as the RTT-RTO algorithm, is shown in Figure 1. Note that for the flow with minimum RTT and RTO, the packets are marked based on the ratio of its RTT and RTO. Otherwise, the right hand side of equation (9) may become 1 and all packets of the flow with minimum RTT will be marked as $DP2$, which will deteriorate the performance of the flow.

V. Simulation Setup

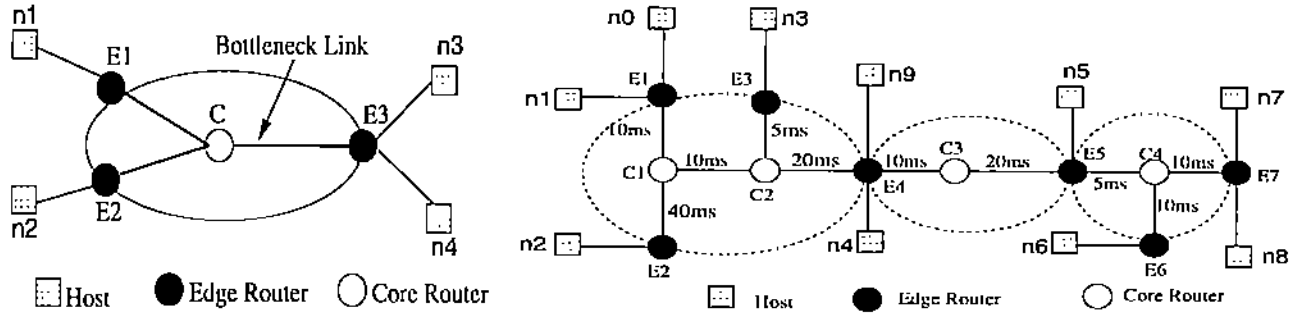
We use the **ns-2** simulator [13] for our experiments. For the standard diff-serv implementation, we use software developed at Nortel Networks [14]. We use the combination of TSW tagger [3], a rate estimator, and the TSW3CM marker [8] to refer as a standard conditioner.

1. If measuredRate <= targetRate
2. mark packets as DP0
3. Else
4. mark packets as DP0
5. Else
6. mark packets as DP0 with probability $(1-p^2)$
7. If packet is not marked DP0
8. mark packets as DP1 with probability (1-q)
9. mark packets as DP2 with probability q
10. where p and q are:
11. $p = \frac{(measuredRate - targetRate)}{measuredRate}$
12. $q = \left(\frac{minRTT}{aggregateRTT}\right)^2 \left(\frac{minRTO}{aggregateRTO}\right)$

Fig. 1. An RTT-RTO aware Traffic Conditioner with three drop precedences.

The simple network topology shown in Figure 2(a) is used to show problems with RTT-aware conditioners and how RTT-RTO and RTT-SW can overcome these problems. We use the multiple domain topology in Figure 2(b) with cross traffic to illustrate more realistic scenarios. Each edge router is connected to a host which sends aggregate flows to simulate different users. The RED parameters $\{min_{th}, max_{th}, P_{max}\}$ used are: for DP0 {40,55,0.02}; for DP1 {25,40,0.05}; and for DP2 {10,25,0.1} as suggested in [4]. w_q is set to 0.002 for all REDs. TCP New Reno is used with a packet size of 1024 bytes and a maximum window of 64 packets.

We use 10 micro-flows (where a micro-flow represents a single TCP connection) per aggregate when we simulate a small number of micro-flows and 200 micro-flows for a large number of flows. The metrics we use to evaluate performance are: **Throughput:** Average (over simulation time) bytes received by the receiver application per second; **Packet Drop Ratio:** Ratio of total packets dropped at the core to the total packets sent; **Packet Delay:** Average delay to deliver a packet for delay sensitive



(a) Simple topology

(b) Multiple domain topology

Fig. 2. Simulation topologies. All links are 10 Mbps.

applications like Telnet; **Response Time:** This is the time between sending a request to a server and receiving the response back from the server.

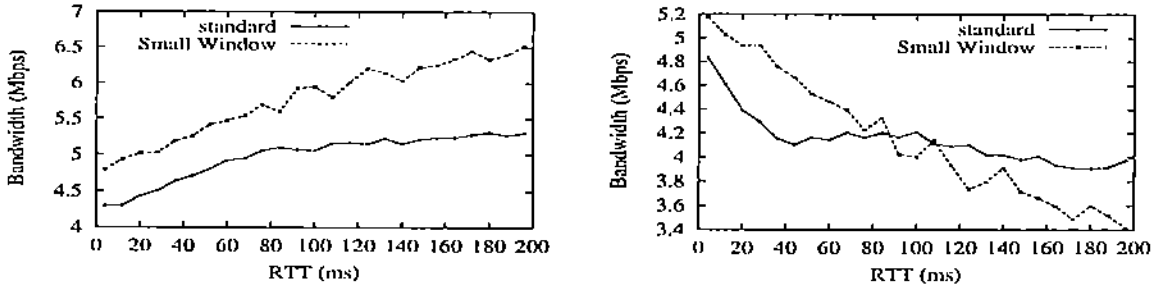
VI. Simulation Results

We first show the behavior of the small window protection marking technique, and then focus on the RTT-RTO aware traffic conditioner. We simulate FTP, Telnet and WWW applications in two topologies.

A. Small Window Protection

The objective of our first experiment is to study the performance of the standard traffic conditioner and show that small window (SW) protection improves performance. We vary the RTTs in this experiment and investigate the effect on throughput and packet drop ratio. We use the simple topology in Figure 2(a) where one aggregate flow, Flow 1, is created between nodes $n1$ and $n3$ with RTT 20 ms and another aggregate flow, Flow 2, is created between nodes $n2$ and $n4$. The RTT of Flow 2 is varied from 4 to 200 ms. Each aggregate flow has a committed rate (CIR) of 2 Mbps and a peak rate (PIR) of 3 Mbps.

Figure 3 shows the bandwidth achieved with and without small window protection in an over-provisioned network. With SW, if the window size of a flow is less than a threshold k, the flow packets



(a) Aggregate Flow 1 (b) Aggregate Flow 2

Fig. 3. Throughput of the standard traffic conditioner with small window. RTT of F1 ($n1 - n3$) is 20 ms and RTT of F2 ($n2 - n4$) is shown on the x-axis.

are marked as DP0. We use $k = 3$ in all our experiments in this paper. Summing up the value in both parts of figure 3, we observe that the total achieved bandwidth with SW is higher than the standard conditioner and is close to the link capacity. SW protection also works well for small and large numbers of micro flows as well as for short and long lived flows, and over and under provisioned networks (detailed results with different SW thresholds and target rates are given in [11]). SW favors short RTT flows (Flow 1). Long RTT flow bandwidth deteriorates because short RTT flows get more protection. We have also observed that the packet drop ratio decreases when the RTT of Flow 2 increases, because for longer RTT, TCP can estimate the sending rate more accurately.

B. RTT-Aware Traffic Conditioners

As previously mentioned, we have observed that a basic RTT-aware conditioner (with both 2 and 3 Drop Precedences) as in [4] is biased when a large number of flows is being multiplexed. Using the same experimental setup as the previous experiment, we observe that Flow 2 (the longer RTT flow) obtains most of the extra bandwidth after target rates have been satisfied for both aggregates. Figure 4 shows that Flow 1 achieves only 2.3 Mbps whereas Flow 2 gets 7.52 Mbps (at Flow 2 RTT=100 ms) with the basic RTT-aware conditioner.

We trace the reason for this behavior to the fact that Flow 2 gets priority over Flow 1 due to its longer

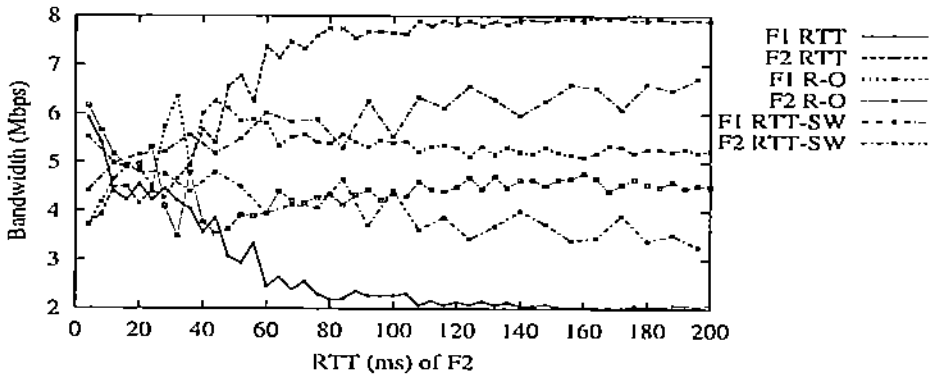


Fig. 4. Throughput comparison of basic RTT, RTT-RTO (R-O), and RTT-SW based conditioners. RTT of F1 ($n1 - n3$) is 20 ms and RTTs of F2 ($n2 - n4$) is shown on the x-axis.

RTT, after target rates are satisfied. As a result, many micro flows in the aggregate Flow 1 time-out, and Flow 1 cannot achieve more than its target rate. Figure 5 shows that the congestion window (*cwnd*) of a randomly selected micro flow in the Flow 1 aggregate remains small due to timeouts. The figure also shows that incorporating small window protection overcomes this problem.

Figure 4 illustrates that our proposed RTT-RTO (R-O) based conditioner (as well as the incorporation of small window protection into the RTT-aware conditioner (RTT-SW)) mitigate this RTT-based unfairness. This is because with a larger number of flows, the per micro flow bandwidth share is small and thus the steady-state *cwnd* is reduced. When *cwnd* is small, there is a higher probability of timeouts in the case of packet drops. Protecting packets (via DP0 marking) when the window is small reduces time-outs, especially back-to-back time-outs. The micro flow also recovers from timeouts when RTO as well as RTT is used to mark packets and fairness is improved.

C. Multiple Domain Topology

To examine more realistic scenarios, we use the multiple domain topology shown in figure 2(b) where flows traverse multiple differentiated services domains. We have created flows $F1 = n1$ to $n7$, $F2{=}n2$ to $n8$, $F3{=}n3$ to $n4$, $F4{=}n0$ to $n9$, and $F5{=}n5$ to $n6$. The first two aggregate flows traverse multiple

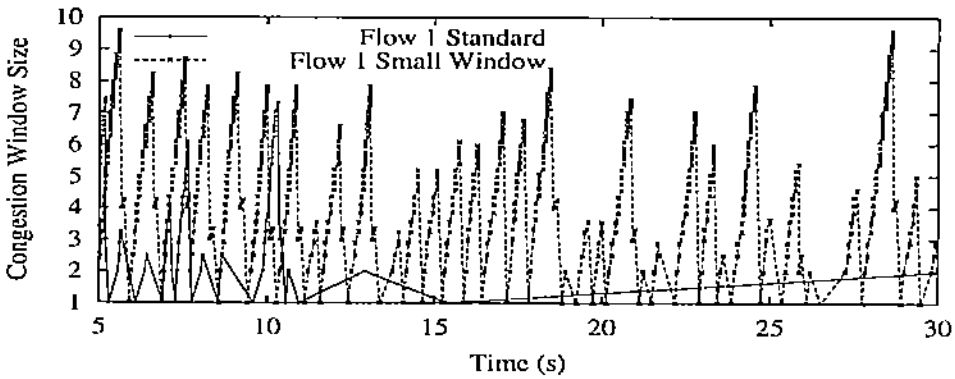
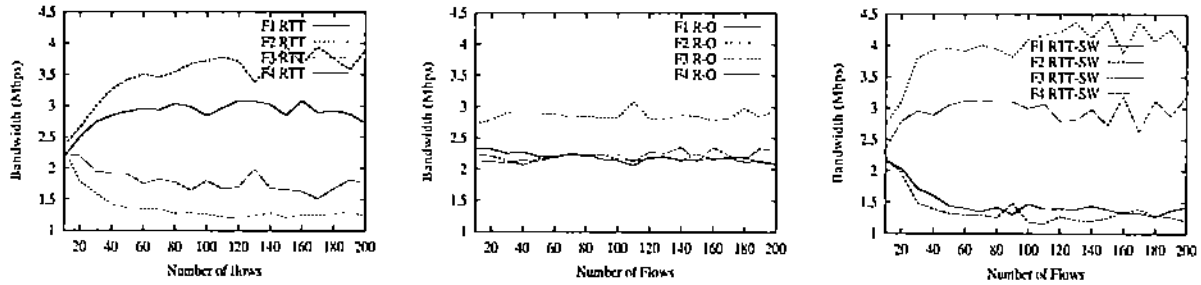


Fig. 5. Congestion window size with and without small window protection with RTT-based conditioners for a micro flow of Flow 1.

domains while the remaining two act as cross traffic. $F1$ and $F2$ have longer RTTs whereas $F3$, $F4$, and $F5$ have short RTTs. Figure 6 shows that, with the basic RTT-aware conditioner, $F1$ and $F2$ obtain much higher bandwidth than flows with short RTTs. We discard initial values to reduce transient effects on the result. With the basic RTT-aware conditioner, the excess bandwidth is distributed according to the RTT so that the longer RTT flows get higher share. We do not see this unfairness with the RTT-RTO conditioner or with RTT-SW. With RTT-SW the short RTT flows get much higher bandwidth than long RTT flows. The RTT-RTO based conditioner is fair because long RTT flows do not get higher bandwidth as with the basic RTT-aware conditioner, but also short RTT flows do not steal most of the resources as with RTT-SW. Flows $F1$, $F2$, and $F4$ achieve almost same amount of bandwidth and flow $F3$ gets little higher, which is fair because this flow has a very short RTT. If the network is extremely over-provisioned, the performance difference is more pronounced. We have observed that flow $F3$ obtains 67 times more bandwidth than what $F1$ and $F2$ achieved with the standard conditioner, whereas with RTT-RTO the flows achieve very similar bandwidths.

D. Telnet and WWW Traffic

We compare the performance of Telnet (delay-sensitive) and WWW (response time sensitive) applications with the various RTT-aware conditioner variations. For the Telnet experiment, the metric



(a) RTT-aware (b) RTT-RTO (c) RTT-SW

Fig. 6. Throughput of RTT-aware traffic conditioners in a multiple domain topology (shown in Figure 2(b)) for various number of micro-flows. F1, F2 are long RTT flows and F3 has very short RTT. F4 is in the middle.

TABLE I

PER TELNET PACKET DELAY (FIRST THREE COLUMNS) AND PER SESSION DELAY FOR TELNET TRAFFIC. NUMBER OF TELNET SESSIONS = 100.

Conditioner	Delay (s) F1, F2, F4	Delay (s) F3, F5	Delay (s) overall	Delay (s) / session
Standard	5.36	2.32	3.62	72.11
Basic RTT	5.23	2.18	3.48	69.19
RTT-RTO	5.32	1.98	3.19	68.68
RTT-SW	5.12	1.84	2.89	66.09

used is the average packet delay for each Telnet packet. The topology is the same as figure 2(b), but all links capacities are set to 1 Mbps to induce congestion. We simulate 100 sessions each from node $F1$=$n1$-$n7$, $F2$=$n2$-$n8$, $F3$=$n3$-$n4$, $F4$=$n0$-$n9$, and $F5$=$n5$-$n6$. Each session transfers less than 10 to more than 30 TCP packets.

Table I shows the average packet delay for each Telnet packet. We compare the standard, the basic RTT-aware conditioner, the RTT-RTO conditioner and the RTT-aware conditioner with small window protection (RTT-SW). The delays are long because the network is congested. The standard conditioner has the highest delay for long RTT flows. The RTT-SW has the lowest delay for short RTT flows. This is because with small window protection, short RTT flows get much better service than the long RTT

flows. With the RTT-RTO conditioner, the delay for long RTT flows is lower than with the standard and RTT-aware conditioners. In some cases, short RTT flows have higher delay with the RTT-RTO conditioner, which is consistent with the fairness objective of the conditioner. Our experiments show that we can achieve better overall performance with the RTT-RTO conditioner because the delay of long RTT flows is reduced with RTT-RTO aware conditioner and the overall Telnet packet delay for all flows is minimized. The per Telnet session delay is low with RTT-RTO conditioner.

As web traffic constitutes most (60%-80%) of the Internet traffic, we examine our traffic conditioners with the WWW traffic model in **ns-2** [13]. Details of the model are given in [15]. The model uses HTTP 1.0 with TCP Reno. Servers are attached to $n4$, $n7$ and $n8$ of Figure 2 (b), while $n1$, $n2$ and $n3$ are used as clients. A client can send a request to any server. Each client generates a request for 5 pages with a variable number of objects (e.g., images) per page. We use the default **ns-2** probability distribution parameters to generate inter-session time, inter-page time, objects per page, inter-object time, and object size (in kB).

Table II shows the average response time per WWW request received by the client for 50 concurrent sessions. The network setup is the same as with Telnet traffic. Two response times are shown in the table: one is the time to get the first response packet and another is to get all data. The table shows that the RTT-RTO conditioner reduces total response time over all other conditioners. The RTT-SW conditioner takes less time for the first packet because of the small window protection at the time of connection setup. For 100 concurrent sessions, RTT-RTO conditioner takes the minimum time to get first response. The response time does not differ significantly if the network is not congested.

VII. Conclusions

In this paper, we have shown that using a basic RTT-aware traffic conditioner can be unfair by giving all extra bandwidth to long RTT flows when many micro-flows traverse through an edge router.

TABLE II

RESPONSE TIME FOR WWW TRAFFIC. NUMBER OF SESSIONS = 50

Conditioner	Avg response time (sec): first packet	Std dev	Avg response time (sec): all packets	Std dev
Standard	0.75	1.60	2.25	4.79
Basic RTT	0.71	1.52	2.16	4.62
RTT-RTO	0.77	1.64	1.69	3.61
RTT-SW	0.64	1.37	1.80	3.83

This behavior causes short RTT flows to starve because they frequently time-out and go to slow start. To overcome this unfairness, we present two schemes: one protects flows with small windows, and the other re-designs the conditioner using both RTT and RTO values. Both conditioners are shown to perform well for both small and large numbers of flows. The RTT-RTO conditioner is shown to improve FTP throughput, reduce packet delay for Telnet and response time for WWW traffic.

We note that when a packet is protected (it is re-marked to green when it was yellow or red), the flow profile must still be preserved by marking later packets yellow or red. This ensures that the congestion situation of the network does not deteriorate due to this flow protection.

RTT-aware conditioners require edge routers to determine the RTT of aggregates passing through them. The RTT can be measured by monitoring the flow sequence number in one direction and observing the ACKs in the other direction. This approximation works because the conditioner compares approximate values to each other. It is possible to take a a single flow as a representative of the aggregate. As an RTT-aware conditioner also requires the minimum aggregate RTT, the edge routers need to exchange this information. The retransmission timeout can be approximated based on the RTT value using the RTT variance. The efficient implementation of RTT-aware conditioners is the subject of our ongoing work.

ACKNOWLEDGMENTS

The authors would like to thank Nabil Seddigh and Peter Pieda from Nortel Networks for their valuable comments.

REFERENCES

[1] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An architecture for Differentiated Services," RFC 2475, December 1998.

[2] S. Floyd and V. Jacobson, "Random Early Detection gateways for congestion avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, 4, pp. 397–413, 1993.

[3] D.D. Clark and W. Fang, "Explicit allocation of best effort packet delivery service," *IEEE/ACM Transactions on Networking*, vol. 6, 4, 1998.

[4] B. Nandy, N. Seddigh, P. Pieda, and J. Ethridge, "Intelligent Traffic Conditioners for Assured Forwarding based Differentiated Services networks," *IFIP High Performance Networking (HPN 2000), Paris*, June 2000.

[5] J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski, "Assured Forwarding PHB group," RFC 2597, June, 1999.

[6] J. Ibanez and K. Nichols, "Preliminary simulation evaluation of an Assured Service," Internet Draft, draft-ibanez-diffserv-assured-eval-00.txt, August 1998.

[7] N. Seddigh, B. Nandy, and P. Pieda, "Bandwidth assurance issues for TCP flows in a Differentiated Services network," *Globecom*, 1999.

[8] W. Fang, N. Seddigh, and B. Nandy, "A Time Sliding Window Three Colour Marker," RFC 2859, June 2000.

[9] M. Mathis, J. Semke, J. Mahdavi, and T. Ott, "The macroscopic behavior of the TCP congestion aviodance algorithm," *ACM SIGCOMM Computer Communication Review*, vol. 27, No. 3, pp. 67–82, 1997.

[10] A. Feroz, S. Kalyanaraman, and A. Rao, "A TCP-Friendly traffic marker for IP Differentiated Services," *Proc. of the IEEE/IFIP Eighth International Workshop on Quality of Service - IWQoS*, 2000.

[11] A. Habib, S. Fahmy, and B. Bhargava, "Design and evaluation of an adaptive traffic conditioner in differentiated serivces networks," *IEEE ICCCN, Scottsdale, Arizona, USA*, Oct 2001.

[12] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP throughput: A simple model and its empirical validation," *ACM SIGCOMM*, 1998.

[13] S. McCanne and S. Floyd, "Network simulator ns-2," http://www.isi.edu/nsnam/ns/, 1997.

[14] F. Shallwani, J. Ethridge, P. Pieda, and M. Baines, "Diff-Serv implementation for ns," http://www7.nortel.com:8080/CTL/#software, 2000.

[15] A. Feldmann, A. C. Gilbert, P. Huang, and W. Willinger, "Dynamics of IP traffic: A study of the role of variability and the impact of control," *ACM SIGCOMM '99*, pp. 301–313, 1999.