Design and Evaluation of an Adaptive Traffic Conditioner for Differentiated Services Networks

Ahsan Habib, Sonia Fahmy, Bharat Bhargava
Center for Education and Research in Information Assurance and Security (CERIAS),
and Department of Computer Sciences
Purdue University, West Lafayette, IN 47907-1398, USA
E-mail: {habib, fahmy, bb}@cs.purdue.edu

Abstract—**We design and evaluate an adaptive traffic conditioner to improve application performance over the differentiated services assured forwarding behavior. The conditioner is adaptive because the marking algorithm changes based upon the current number of flows traversing through an edge router. If there are a small number of flows, the conditioner maintains and uses state information to intelligently protect critical TCP packets. On the other hand, if there are many flows going through the edge router, the conditioner only uses flow characteristics as indicated in the TCP packet headers to mark without requiring per flow state. Simulation results indicate that this adaptive conditioner improves throughput of data extensive applications like large FTP transfers, and achieves low packet delays and response times for Telnet and WWW traffic.**

I. Introduction

The differentiated services (diff-serv) architecture [1] is a simple and scalable approach to improve Quality of Service (QoS) for data and multimedia applications in IP networks. The diff-serv model uses Traffic Conditioners (TCs) at the edges of an administrative domain to shape, mark, and drop traffic if necessary. The operations are based on Service Level Agreements (SLAs) between adjacent domains. In the core of the network, Per Hop Behaviors (PHBs) are used to achieve service differentiation by intelligently dropping and scheduling packets according to their markings. The current diff-serv model defines two forwarding mechanisms: Expedited Forwarding and Assured Forwarding (AF). For the AF service, core routers use an active queue management technique such as Random Early Detection (RED) [2] with multiple thresholds such as RIO (RED with IN/OUT) [3].

Designing an edge router that intelligently conditions AF traffic has been an active research area. Several studies show that application performance is poor if traffic conditioning at network edges does not consider transport protocol reaction to drop at the end systems, and dropping behavior at the core routers. Several proposals adjust the marking, dropping, or shaping scheme of a traffic conditioner based upon TCP connection state. Most of these proposals, however, do not scale well to large numbers of flows. In addition, the proposals only consider bulk data applications and do not examine delay-sensitive traffic and WWW traffic.

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

We study the behavior of transport protocols and use TCP characteristics to develop an adaptive traffic conditioner that protects critical TCP packets from drop in order to avoid TCP timeouts. Each conditioner feature is studied individually and then they are studied in combination. Our conditioner behaves differently based on the number of flows traversing it. This adaptive design overcomes scalability problems arising from maintaining excessive per flow state. The performance of the conditioner is analyzed both for data intensive applications and delay sensitive applications in multiple-domain, variable delay configurations.

The remainder of this paper is organized as follows. Section II presents the basics of traffic conditioning and differential drop. Section III discusses previous work on diff-serv conditioner design. Section IV discusses features of TCP-sensitive conditioners and how to combine them in our proposed adaptive traffic conditioner. Section V contains the details of our simulation setup. Section VI presents and discusses the simulation results. We conclude with a summary and discussion of future work.

II. Background

This section describes the components of a traffic conditioner and how they relate to differential drop at core routers.

A. Basics of a Conditioner

A traffic conditioner may contain meters, markers, droppers, and shapers for conditioning functions [1]. The conditioner may re-mark a traffic stream or discard or shape packets to alter the temporal characteristics of the stream and bring it into compliance with a traffic profile specified by the network administrator. As shown in Figure 1, incoming traffic passes through a classifier, which is used to select a class for each traffic flow. The meter measures and sorts the classified packets into precedence levels. The decision (marking, shaping, or dropping) is made based on the measurement result.

Assured forwarding provides up to three drop precedences for each queue[4]. We assume the drop precedences are DP0, DP1 and DP2, where DP0 means lower precedence to drop, and DP2 means higher. The Differentiated Services Code Point (DSCP) (contained in the IP header DSFIELD/ToS) is set to mark the

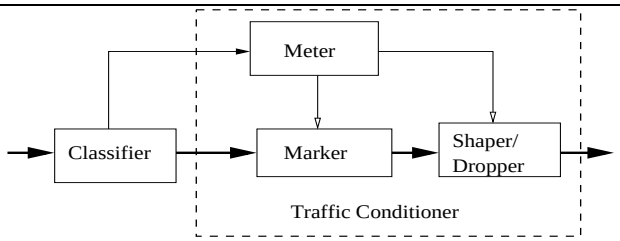


Fig. 1. Components of a Traffic Conditioner

DP. When congestion occurs, packets marked with DP2 have to be dropped, followed by DP1, and then DP0.

Shaping the traffic reduces the traffic variation and makes it smooth. It also provides an upper bound for the rate at which the flow traffic is admitted into the network.

B. Differential Drop

Within each core assured service queue, discrimination among packets is done using a differential drop algorithm. Different queue mechanisms can be used to realize this preferential drop.

The RIO algorithm distinguishes between two types of packets, IN and OUT of profile, using two RED instances. Each RED instance is configured with min_{th}, max_{th}, and P_{max}. Suppose the parameters for the IN profile packets are min_{in}, max_{in}, and P_{max_in}, and for the OUT of profile packets are min_{out}, max_{out}, and P_{max_out}. To drop OUT packets earlier than IN packets, min_{out} is chosen smaller than min_{in}. The router drops OUT packets more aggressively by setting P_{max_out} higher than P_{max_in}. To realize three drop precedences, three REDs can be used. The Assured Forwarding PHB provides four classes (queues) of delivery for IP packets and three levels of drop precedence per class. The average queue size is calculated using an exponentially weighted moving average algorithm with parameter w_q.

III. Related Work

The assured forwarding behavior has been extensively studied in the last few years. Clark and Fang introduced RIO in 1998 [3], and developed the Time Sliding Window (TSW) tagger. The TSW tagger provides a smooth estimation of the TCP sending rate. They show that sources with different target rates can achieve their targets using RIO even for different Round Trip Times (RTTs), whereas simple RED routers cannot.

Ibanez and Nichols [5] used a token bucket marker for Assured Service and showed that target rates and TCP/UDP interaction are key factors in determining throughput of flows. The TCP response to packet loss is the main problem. Seddigh, Nandy and Pieda [6] showed that the above mentioned factors are also critical to the distribution of excess bandwidth in an over-provisioned network. Lin, Zheng and Hou [7] proposed an enhanced TSW profiler, but their solution requires state information to be maintained at core routers, which does not scale well. Fang, Seddigh and Nandy [8] proposed the Time Sliding Window Three Color Marker (TSW3CM), which we use as a standard conditioner.

Yeom and Reddy [9] pass the marking information to the sender, so that a sender can slow down its sending rate in the case of congestion. This requires modifying the host TCP implementation. They also use three drop precedences IN, OUT-IN and OUT-OUT to provide better QoS. Storing and searching per flow information at the border router for a large number of flows may, however, not scale well.

Feroz et al. [10] propose a TCP-Friendly marker. As TCP applications are influenced by bursty packet loss behavior, they use TCP characteristics to design their marker. The main concept is to "protect small-window flows from packet losses" by marking their traffic IN. Detailed analysis on a good window size threshold (below which a flow is marked as IN) for various situations is not provided in [10]. We investigate different thresholds to identify a small window and analyze how they affect the throughput of flows with different RTTs.

Another adaptive packet marker proposed by Feng et al. [11] uses a Packet Marking Engine (PME), which can be a passive observer under normal conditions, but becomes an active marker at the time of congestion. The marking rate is adjusted by the throughput. This engine can be source transparent or source integrated. The host TCP reacts to the marked/unmarked packet drop differently using two congestion windows: one for best effort traffic and another for priority traffic. The source integrated technique is hard to deploy.

IV. Proposed Traffic Conditioner

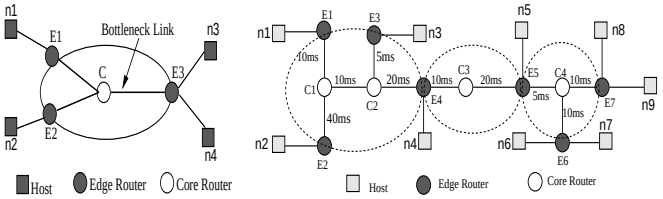
In this section, we discuss techniques to incorporate in a conditioner to improve performance of applications running on top of TCP. Some of these techniques are (loosely or closely) based on ideas proposed in the literature as cited below, but the techniques not requiring per flow state, the combination of techniques, and the adaptivity of the conditioner to the number of flows have not been previously proposed. We use the TSW tagger [3], a rate estimator, and the TSW3CM marker [8] as a standard conditioner [12]. In addition, we examine the following TCP-adaptive features:

SYN: The first few packets of a TCP flow should not be dropped to allow the TCP congestion window to grow. At the edge router, the first few packets can be identified by their sequence numbers. As the initial TCP sequence number is not known to the conditioner, the conditioner needs to store it. To avoid storing per flow information at the edge, we propose to give low drop priority only to SYN packets as indicated in the TCP header

Small Window (SW): We protect small window flows from packet losses by marking them with DP0. This strategy was introduced in [10]. 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. We give low drop priority to flows with small congestion window sizes. The calculation of

```
For each incoming flow
    If there is a state entry for this flow
        statePresent = TRUE
        Update the state table to reflect recent information
    Else
        statePresent = FALSE
        Add the flow in the state table, overwrite if table is full
    If statePresent is TRUE
        Use Standard Traffic Conditioner with SYN, CWR, SW, Burst
    Else
        Use Standard Traffic Conditioner with SYN and CWR
```

Fig. 2. Algorithm for Adaptive Traffic Conditioner



(a) Simple topology (b) Multiple domain topology

Fig. 3. Simulation topologies. All links are 10 Mbps.

TCP window size needs sequence number of data and acknowledgment (ACK) packets. This technique requires per flow state at the edge router. We use SW when we have state information and use SYN otherwise.

Congestion Window Reduction (CWR): ECN-Capable TCP may reduce its congestion window due to a timeout, triple duplicate ACKs, or in response to explicit congestion notification (ECN) [13]. TCP sets the CWR flag in the TCP header of the first data packet sent after the window reduction. The CWR bit should not be set on retransmitted packets [14] for greater robustness and against denial of service attacks. Instead, when the TCP data sender is ready to set the CWR bit after reducing the congestion window, it should set the CWR bit on the first new data packet that it subsequently transmits. We propose to give low drop priority for a packet if the CWR bit is set. This avoids consecutive *ssthresh* reductions that lead to poor performance with TCP Reno [15].

Target Rate (TR): The target rate is an important factor in marking. Nandy et al. [12] mark DP1 and DP2 only when target rates have been achieved, and marking is inversely proportional to the square of the flow requested rates if proportional sharing of excess bandwidth is required. Another strategy is to mark packets based on the difference between target rate and exponentially averaged input rate of the aggregate in order to improve fairness. We use the first strategy.

Burst: The marker avoids marking high drop priority in bursts to work well with TCP Reno. The shaper avoids burstiness to avoid consecutive packet drops and poor performance. This strategy was introduced in [10].

Each of the above techniques has advantages and limitations. SYN, CWR, and aggregate Target Rate do not need to store per flow information and are simple to implement. On the other hand, SW, Target Rate based on individual information, and Burst need to maintain and process per flow information. Storing and processing excessive state information about each micro-flow at the edge does not scale well. To overcome this problem, we propose adaptive conditioning. In our design, the edge router stores per flow information for a certain number of flows based on its available resources. If its per flow state table for N flows is full, the router overwrites previous entries. N here is a router configuration parameter that depends on router memory size. We choose to maintain state for the N most recent flows, thus implementing a least recently used (LRU) replacement strategy.

For an incoming flow, conditioning is done based on state information if it is present. If there is no state present, conditioning only uses techniques that do not need per flow information. In this way, the router does not handle state information beyond its capabilities and avoids scalability problems. The conditioner algorithm is given in Figure 2.

V. Simulation Setup

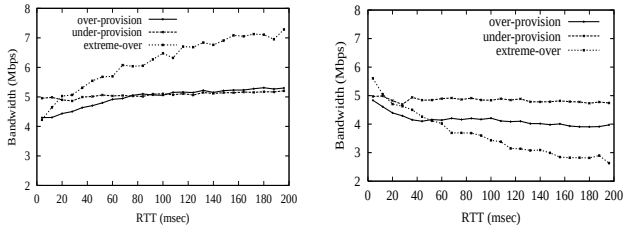
We use the **ns-2** simulator [16] for our experiments. For the standard diff-serv implementation, we use software developed at Nortel Networks [17].

The simple network topology shown in Figure 3(a), is used to test the different marking techniques discussed in section IV. We also use the multiple domain topology in Figure 3(b) to evaluate our proposed conditioner. 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 by [12]. w_q is 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 transmit a packet for delay sensitive applications like Telnet; **Response Time:** This is the time between sending a request to a web server and receiving the response back from the server.

VI. Simulation Results

We study the behavior of the standard traffic conditioner and each marking technique individually and in combination. We also study the performance of the proposed adaptive traffic conditioner with FTP, Telnet and WWW applications. Network hosts and routers are ECN-enabled for all experiments. Some data/graphs are not presented in this paper due to space limitation. Please see [18] for details.



(a) Aggregate Flow 1-3 (b) Aggregate Flow 2-4

Fig. 4. Throughput for standard traffic conditioner in over, under and extremely over-provisioned networks for 200 flows.

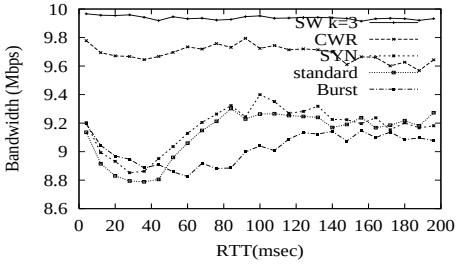


Fig. 5. Throughput comparison of the standard traffic conditioner and various marking techniques with 200 flows.

A. Marking Techniques

The objective of our first experiment is to study how each marking technique discussed in section IV affects the performance of the standard traffic conditioner individually and collectively. We vary the RTTs, window size for SW, and target rates in this experiment. The output parameters (metrics) are throughput and packet drop ratio. We use the simple topology in Figure 3(a) where one aggregate flow, Flow 1-3, is created between nodes $n1$ and $n3$ with RTT 20 ms and another aggregate flow, Flow 2-4, is created between nodes $n2$ and $n4$. The RTT of Flow 2-4 is varied from 4 to 200 ms.

Standard Conditioner: We test the conditioner for both small (10 micro flows) and large (200 micro flows) number of flows, in under and over-provisioned networks. All flows have the same target rate. For the over-provisioned case, the committed rate, CIR, is 2 Mbps and peak rate, PIR, is 3 Mbps for each aggregate flow. For the extremely over-provisioned case, CIR is 0.2 Mbps and PIR is 0.3 Mbps, and for the under-provisioned one, CIR is 6 Mbps and PIR is 10 Mbps.

Figure 4 shows the achieved bandwidth for the under, over and extremely over-provisioned network cases as RTT of Flow 2-4 varies. In the under-provisioned case, both flows achieve close to 5 Mbps, which is a desirable outcome. In the over-provisioned cases, small RTT connections are favored. For example, Flow 1-3 is favored at the expense of Flow 2-4 when its RTT is lower than the RTT of Flow 2-4 (when RTT on the x-axis is greater than 20 ms). If the network is extremely over-provisioned, we see more unfairness and higher packet drop ratio. This is because TCP connections are very aggressive for the flow with small RTT. Due to the fluctuation of the sending rate, TCP loses more packets. As the RTT of Flow 1-3 is fixed, it has almost the same packet drop ratio throughout the the experiment, but the drop ratio decreases when RTT of Flow 2-4 increases. This is because for higher RTT, TCP can estimate the sending rate more accurately.

SYN: SYN is useful for short-lived connections and high degrees of multiplexing. Even though the bandwidth improvement we observed in experiments is insignificant (200 kbps for the total), SYN can be used when other expensive techniques (in terms of complexity to deploy) cannot be used. This technique is used in the adaptive conditioner when there is no state information about the flow.

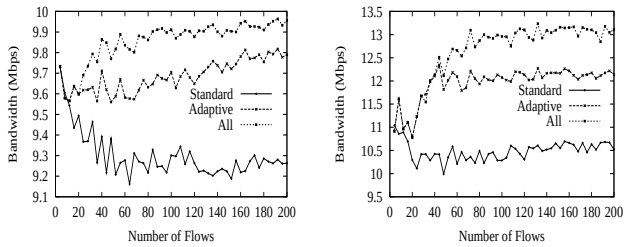
Small Window: Small window (SW) works both for small and large number of micro flows as well as short and long lived flows. To study the effect of the window size, *k*, on achieved bandwidth on both flows, *k* is varied from 3 to 10. If the window size of a flow is less than *k*, the flow packets are marked DP0. We have observed that the larger value of *k* helps the (more aggressive) small RTT connection (Flow 1-3) to achieve more bandwidth at the expense of the large RTT flow (Flow 2-4) due to the preferential drop at the core. This contrast is clearer in an under-provisioned network. The total achieved bandwidth is higher than the standard conditioner and is close to the link capacity. Thus, SW significantly improves utilization. The choice of *k* depends on policy. A higher value of *k* such as 7 or 8 may favor short RTT flows and result in more unfairness against long RTT flows, while a lower value of *k* (e.g., 3) avoids this problem.

Congestion Window Reduction (CWR): Giving priority to CWR packets helps the growth of the congestion window after reductions and reach equilibrium. Results show that CWR helps Flow 2-4 to achieve higher throughput. Flow 1-3 sometimes times out and has high packet drop.

Burst: Avoiding bursty marking and shaping packet bursts improves achieved bandwidth over the standard traffic conditioner. The improvement is more significant for both flows when RTT is low. Flow 2-4 achieves its highest bandwidth in an over-provisioned network when Burst and CWR are combined for low RTT. The "Burst" technique exhibits the lowest packet drop ratio for both flows among other techniques when each is studied separately.

Target Rate: We use a Target Aware traffic conditioner to divide excess bandwidth in an over-provisioned network in proportion to the subscribed target rates [12]. This feature has no effect in cases of congestion.

Combinations and Overall Performance: Figure 5 compares different marking techniques in separate simulation runs. From the figure, it is clear that the "small window" technique contributes most to total bandwidth gain, followed by CWR and SYN. SW favors short RTT connections (Flow 1-3), but it reduces packet drop ratio and timeouts for Flow 2-4 as well, compared to the standard traffic conditioner. "Burst" is effective for short RTT (less than 40 ms). If SW is not used, Burst+CWR achieves higher bandwidth than any other combination. Although SW works better than any other technique alone, using all design techniques together has advantages over SW alone.



(a) Simple Topology (b) Multi-Domain Topology

Fig. 6. Achieved bandwidth by the standard conditioner, adaptive conditioner, and standard conditioner with all techniques. a) state table size=20 micro-flows b) State table size=50 micro-flows

As previously discussed, in an under-provisioned network SW increases the throughput of Flow 1-3 at the expense of Flow 2-4. Fluctuations occur when RTT is relatively low for both connections. The fluctuations can be overcome by using the Burst technique. CWR helps Flow 2-4 to achieve more bandwidth as before.

B. Adaptive Conditioner

We examine our proposed adaptive conditioner for both the simple and multiple domain topologies. The algorithm used for conditioning is described in section IV. Figure 6(a) compares achieved bandwidth with the standard, adaptive, and the standard conditioner with all marking techniques at the same time (referred as "All") for the simple topology shown in Figure 3(a) with different number of flows. The adaptive conditioner switches the marking techniques based on the availability of state information but "All" has a huge state table so that it can use all marking techniques described in Section IV to mark every packet. The adaptive conditioner outperforms the standard one for both aggregate flows. The adaptive conditioner is more fair in the sense that Flow 1-3 does not steal bandwidth from flow 2-4, and total achieved bandwidth is close 10 Mbps (bottleneck link speed). Aggregate Flow 2-4 performs better in the case of the adaptive conditioner than when using "All."

Figure 3(b) shows our second simulation topology. Three domains are interconnected, with all links being 10 Mbps. We create aggregate flows between nodes $n1$-$n8$, $n2$-$n9$, $n3$-$n4$, $n5$-$n6$, and $n7$-$n9$. Flows have very different RTTs and bottlenecks. Not all flows start/stop transmission at the same time. Short-lived flows last from less than a second to a few seconds. Flows from multiple hosts sometimes traverse the same edge router. $C2$-$E4$, $E5$-$C4$ and $C4$-$E7$ are the busiest links. We measure the total throughput over the simulation time at the receiving end. Figure 6(b) shows the total bandwidth gain for this topology comparing different conditioners. From the figure, the adaptive conditioner works better than the standard one and achieves performance close to the complex conditioner ("All") avoiding its scalability problems. The adaptive conditioner improved throughput over the standard conditioner, and improves fairness between low and high RTT flows, without requiring large per-micro-flow state tables.

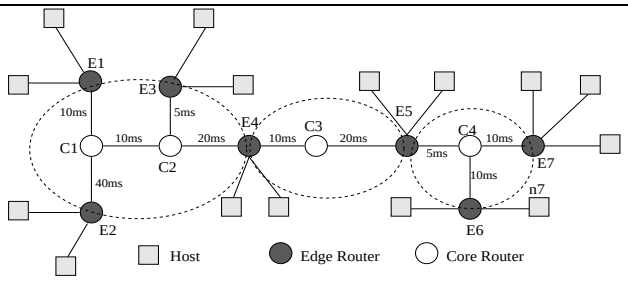


Fig. 7. Complex Multiple Domain Topology

Micro flows	Standard BW gain	Adaptive BW gain	Adaptive (% flows covered at E4)	All techniques BW gain
10	12.65	12.87	41.16	12.87
50	12.18	13.84	16.66	14.20
100	11.67	13.48	8.33	14.89
200	11.77	13.61	4.16	14.91

TABLE I

PERFORMANCE FOR TOPOLOGY IN FIGURE 7. BANDWIDTH (BW) SHOWN IS IN MBPS. STATE TABLE SIZE = 50 MICRO-FLOWS.

Figure 7 shows a variation on the multiple domain topology, where many hosts are connected at most edge routers. The link delay between host and the edge is varied from 1 to 10 ms for different hosts connected to a domain to simulate users at variable distances from same edge routers. This topology is more realistic and some edges such as $E4$ experience a large number of micro-flows. Since each aggregate flow contains 200 micro flows, the soft state table for the adaptive conditioner covers only a small percentage of the flows passing through it (we use a table for the 50 most recent micro-flows). Table I shows that the bandwidth achieved with the adaptive conditioner is close to the conditioner that uses "All" techniques, and the adaptive conditioner always outperforms standard conditioner.

C. Telnet and WWW Traffic

We compare the performance of Telnet (delay-sensitive) and WWW (response time sensitive) applications with the standard conditioner and adaptive conditioner. For the Telnet experiment, the metric used is the average packet delay time for each Telnet packet. The topology is Figure 3(b), where capacity of $C1$-$E4$ and $E5$-$E7$ links is changed to 0.5 Mbps and all others to 1 Mbps to introduce congestion. We simulate 100 Telnet sessions among hosts $n1$-$n8$, $n2$-$n9$, $n3$-$n4$, $n5$-$n6$, and $n7$-$n9$. A session transfers less than 10 to more than 30 TCP packets.

Table II shows the average packet delay in sec per Telnet packet transfer for all flows. The standard conditioner shows the lowest average packet delay for long RTT flows and the highest delay for short RTT flows than other conditioners. "All" favors short RTT flows mainly because of small window protection. The adaptive conditioner is in the middle of two other conditioners. The overall delay is reduced in the adaptive conditioner than standard one. The adaptive conditioner increases the volume of Telnet packet transfer for short RTT flows than the standard one. The standard deviation of average packet delay follows the same pattern among all conditioners i.e. the standard deviation is high

Conditioner	Long RTT Flows (n1-n8, n2-n9)	Short RTT Flows (n3-n4,n5-n6,n7-n9)	Overall
Standard	9.16	4.86	6.02
Adaptive	11.66	4.52	5.37
All	12.60	4.05	4.65

TABLE II

AVERAGE DELAY IN SEC PER TELNET PACKET TRANSFER FOR TOPOLOGY IN FIGURE 3(B). NUMBER OF TELNET SESSIONS = 100.

Conditioner	Avg response time (sec), first pkt	Std dev	Avg response time (sec), all pkts	Std dev
Standard	0.48	0.17	2.23	0.78
Adaptive	0.45	0.14	2.15	0.75
All	0.49	0.19	2.15	0.71

TABLE III

RESPONSE TIME FOR WWW TRAFFIC. NUMBER OF SESSIONS = 50

when the delay is high and it is low when the delay is low.

As web traffic constitutes most (60%-80%) of the Internet traffic, we test our traffic conditioner with the WWW traffic model in **ns-2** [16]. (Details of the model are given in [19].) The model uses HTTP 1.0 with TCP Reno. Servers are attached to $n6$, $n8$ and $n9$ of Figure 3 (b), while $n1$, $n2$ and $n5$ 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 III shows the average response time per WWW request received by the client. The network setup is same as with Telnet traffic. Two response times are shown in the table; one is to get the first packet and another is to get all data. The table shows that our conditioner reduces response time over the standard traffic conditioner. The adaptive conditioner does not change the response time significantly if the network is not congested.

VII. CONCLUSIONS

In this paper, we have discussed techniques to design TCP-aware traffic conditioners and analyzed the performance of a proposed adaptive conditioner for various flow RTTs and degrees of multiplexing, as well as for over and under-provisioned, single domain and multiple domain, networks. All marking techniques examined improve performance, but the small window (SW) protection contributes the most. Small window, however, appears to mostly favor small RTT flows in our experiments. A lower threshold for the window size reduces this unfairness, without compromising the total bandwidth gain. Congestion Window Reduced (CWR) packet protection favors long RTT flows, while burst avoidance (Burst) is effective when round trip time is small.

We protect flows by giving priority to their critical packets. If 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.

We design an adaptive conditioner which stores state information based on available resources. When the per-micro-flow state table is full, the conditioner overwrites previous state information based on a least recently used strategy. Marking is based on information in packets, such as SYN and CWR, if state information is not available. The proposed adaptive conditioner has been shown to improve FTP throughput, reduce packet delay for Telnet and response time for WWW traffic.

This conditioner can be used for any architecture that supports service differentiation, or even with active queue management techniques at network routers. For example, the RED algorithm at network routers can itself protect critical packets such as SYN and CWR packets without requiring any additional state. Alternatively, the adaptive conditioner concept can be employed with algorithms like RED. The router would, in this case, store state for the most recently seen flows and use this information to make intelligent dropping decisions.

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 Transactions on Networking*, vol. 1, 4, 1993.

[3] D.D. Clark and W. Fang, "Explicit allocation of best effort packet delivery service," *IEEE/ACM Transactions on Networking*, vol. 6, 4, pp. 362–374, 1998.

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

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

[6] N. Seddigh, B. Nandy, and P. Pieda, "Bandwidth assurance issues for TCP flows in a Differentiated Services network," *Globecom 99*, 1999.

[7] W. Lin, R. Zheng, and J. Hou, "How to make Assured Services more assured," *In Proceedings of ICNP*, Oct 1999.

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

[9] I. Yeom and N. Reddy, "Realizing throughput guarantees in a Differentiated Services network," *IEEE Int. Conf. on Multimedia Comp. and Systems*, June 1999.

[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] W.C. Feng, D. Kandlur, D. Saha, and K.G. Shin, "Understanding and improving TCP performance over networks with minimum rate guarantees," *IEEE Transactions on Networking*, vol. 7, 2, pp. 173–186, 1999.

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

[13] K. Ramakrishnan and S. Floyd, "A proposal to add Explicit Congestion Notification (ECN) to IP," RFC2481, January 1999.

[14] S. Floyd and K. K. Ramakrishnan, "TCP with ECN: The Treatment of Retransmitted Data Packets ," draft-floyd-tcp-ecn-00.txt, Oct 2000.

[15] S. Fahmy, *New TCP standards and flavors, High Performance TCP/IP networking*, chapter 13, Prentice Hall, Inc., 2001.

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

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

[18] A. Habib, S. Fahmy, and B. Bhargava, "Adaptive Traffic Conditioner Design in Differentiated Services Networks," Technical Report CSTR-015, Department of Computer Science, Purdue University, Nov 2000.

[19] 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.