

A Method for estimating non-responsive traffic at a router *

Zhili Zhao	Jayesh Ametha, Swaroop Darbha	A. L. Narasimha Reddy
Dept. of Elec. Engg	Dept. of Mech. Engg.	Dept. of Elec. Engg.
214 Zachry	313 EPB	214 Zachry
College Station, TX 77843-3128	College Station, TX 77843-3123	College Station, TX 77843-3128

Abstract

In this paper, we propose a scheme for estimating the proportion of the incoming traffic that is not responding to congestion at a router. The proposed scheme utilizes queue length history, packet drop history, expected TCP and queue dynamics to estimate the proportion. We show that the proposed scheme is effective over a wide range of traffic scenarios through ns-2 based simulations. We also discuss potential applications of such a tool in traffic engineering and control.

1 Introduction

Measurements at various internet sites [1] suggest that 85-90% of current network traffic is based on TCP. However, with the emergence of new applications, the usage of protocols other than TCP is expected to increase. For instance, several multimedia applications rely on UDP to transport packets. Recently, there has been increased interest in developing protocols that respond to congestion differently from TCP and provide smoother bandwidths to end applications [2, 3]. Furthermore, increases in bandwidth and computation power are expected to fuel the growths of multimedia applications that do not rely on TCP. These trends point to increased diversity of network protocols and changes in the distribution of bandwidth among protocols in the future.

One can expect such new applications not to respond properly to network congestion. The impact of such non-responsive applications on the TCP-based applications and the overall goodput of network is of considerable importance and has attracted the attention of researchers [4]. It is also possible to stage denial-of-service (DOS) attacks on end hosts and the network by pumping large amounts of non-responsive traffic into the network. Some of the recent DOS attacks have used such "UDP floods". If the network could regulate the utilization of such traffic to a fraction of the link capacity, the impact of such attacks could be mitigated.

*This work was supported in part by an NSF Career Award and an NSF Grant ANI-9909229

If all the non-responsive traffic used UDP for transport, a simple counter of UDP bytes at a link could give us an estimate of the non-responsive traffic. However, non-responsive traffic may use other protocols or use variants of TCP and some UDP applications, such as Real Audio/Video, actually respond to the congestion of network by adapting the sending rate. We also consider a scenario where the arriving traffic consists of a large number of “small” bandwidth TCP flows. Those flows send out a small amount of bytes intermittently. Usually, they are referred as “web mice” [5]. In this scenario, although each flow employs TCP, the aggregate behavior of those flows at the router can not be differentiated from that of non-responsive traffic. If we employ a protocol counter to count UDP bytes, it yields a count of zero, while the apparent behavior is close to that of non-responsive traffic. As a result, a simple counter of UDP bytes may not suffice.

It is necessary to find mechanisms to estimate the amount of arriving traffic at a router that is not responding to congestion. Such estimations may lead to better control of traffic at a router through appropriate adaptation of traffic control algorithms at a router. For example, estimations of non-responsive traffic may help us choose appropriate parameters for active queue management. Need for such tuning is cited recently in [6]. It has been pointed out by many researchers that this increased diversity leads to different choices for traffic management and congestion control [4, 7, 8]. If we could estimate the amount of non-responsive traffic at the router, it is expected that appropriate traffic control functions can be employed in different operating domains of network traffic.

In this paper, we present a technique for estimating the fraction of the arriving traffic at a router that is non-responsive to congestion. The presented scheme utilizes queue length history and packet drop history at the router and expected TCP response to packet drops, which are easily measured at a router.

Models for TCP dynamics and control have been proposed [9, 10, 14]. These studies have led to better analysis of TCP behavior and proposals for improved traffic controllers [11, 12]. Our estimation algorithm is based on those models. We extend these models to account for heterogeneity in traffic, by including the dynamics of non-responsive flows into our model. Our estimation techniques utilize these traffic models and control theoretic estimation techniques based on packet drop history and queue length history at the router. Based on only these measurable quantities, it is possible to estimate the fraction of arriving traffic that is not responding to congestion. In the estimation, we have used a normalized gradient algorithm on the linearized, extended model of traffic flow. We show its effectiveness through ns-2 [13] based simulations.

The rest of the paper is organized as follows. In Section 2, we present a general model of traffic mix in network. In Section 3, we develop an algorithm for estimation. In Section 4, we present our implementation of the algorithm in ns-2, some simulation results and analysis. In section 5, we discuss the possible applications of the presented technique, its limitations and directions for future work. Section 6 concludes the paper.

2 Model of Traffic Mix

Network traffic consists of responsive flows and non-responsive flows. Those flows are based on TCP, UDP or other protocols. Without loss of generality, we use TCP flows as responsive flows and Constant Bit Rate(CBR) flows as non-responsive flows to describe our model of traffic mix. Models of TCP dynamics have been proposed and studied in [9, 10, 14]. Models are described by either the evolution of window size or the evolution of transmission rate.

In fluid-based models of [9] based on the evolution of window size, AIMD behavior of TCP flow i is described by

$$dW_i(t) = \frac{dt}{R_i(q(t))} - \frac{W_i(t)}{2}dN_i(t). \quad (1)$$

The first term describes the additive increase part and the second term describes the multiplicative decrease part. $N(t)$ is a Poisson process describing packet loss. Queue dynamics is described by

$$\frac{dq(t)}{dt} \approx -C + \sum_{i=1}^N \frac{W_i}{R_i(q)}. \quad (2)$$

The first term describes the decrease of queue length due to packet servicing at output link. The second term describes the increase of queue length due to incoming packets from N TCP flows.

Following the fluid-based models of TCP flows and queue dynamics in [9], we introduce a similar model for both TCP and non-responsive flows over a single bottleneck link by adding differential equation of non-responsive flow in the model and adding non-responsive traffic into total incoming traffic in equation of queue dynamics. Note that we use CBR flows to model non-responsive flows. The window of non-responsive flow ($=$ *sending rate* \times *a fixed measuring period*) is not changing with time. The model for traffic mix is described by

$$\dot{W}_u = 0 \quad (3)$$

$$\dot{W}_s(t) = \frac{1}{R(q(t))} - \frac{W_s(t)W_s(t - R(q(t)))}{2R(q(t))}p(t - R(q(t))) \quad (4)$$

$$\dot{q} = \frac{N_s W_s + N_u W_u}{R(q(t))} - C \quad (5)$$

In eqn. 3, W_u is the window size of a representative non-responsive flow. In eqn. 4, $W_s(t)$ is the window size of a TCP flow; N_s is the number of incoming TCP flows; N_u is the number of incoming non-responsive flows; $R(t)$ is the Round Trip Delay, which is given by $\frac{q(t)}{C} + T_p$, where T_p is a fixed propagation delay; $p(t)$ is the packet drop probability. In eqn. 5, $q(t)$ is the queue length and C is the outgoing link capacity.

Eqn. 3 describes the behavior of a non-responsive CBR flow as an example of non-responsive traffic. As we will show in the next section, the non-responsive traffic behavior with respect to time, described by \dot{W}_u in our model, does not impact our estimation algorithm. As a result, even Variable Bit Rate (VBR) non-responsive flows can be included in our estimation algorithm. Eqn. 4 shows the behavior of a TCP flow in congestion control phase. It is the same as that of the fluid-based model [9]. It indicates that development of window size is related to round trip delay, drop probability and history of window size. Eqn. 5 shows the dynamic of queue length. We make the following simplifying assumptions to make the model easily presentable: (a) consider the traffic model over a single bottleneck link, (b) all TCP flows have same window-adaptation behavior and round trip delay, (c) all non-responsive flows have the same window size during the sampling period. As we will show later through simulations, these assumptions do not limit the application of the model to more general situations. They are primarily made to keep the estimation algorithm easily understandable.

It is observed that the impact of non-responsive traffic is modeled through the impact on queue lengths and hence on round trip delays and packet drop probabilities. Round trip delays and packet drop probability impact the behavior of TCP flows and hence their interaction is captured in our model. Based on the model, we propose an algorithm to find out the proportion of the arriving traffic at a router that is non-responsive.

3 Estimation Algorithm

For the purpose of developing an estimation algorithm, we will express the model, presented in Section 2, in terms of the total responsive load $z(t)$ and the total non-responsive load D . The terms $z(t)$ and D are given by the following relationships:

$$z(t) := N_s W_s(t) \tag{6}$$

$$D := N_u W_u \tag{7}$$

The terms $z(t)$ and D are scaled loads, scaled relative to $R(t)$. The quantity of D will be estimated in the algorithm.

In terms of $z(t)$ and D , the dynamics of traffic mix is given by:

$$\dot{z}(t) = \frac{N_s}{R(t)} - \frac{z(t)z(t-R(t))}{2N_s R(t)} p(t-R(t)) \tag{8}$$

$$\dot{q}(t) = \frac{z(t)+D}{R(t)} - C \tag{9}$$

$$R(t) = \frac{q(t)}{C} + T_p \tag{10}$$

where $\dot{z}(t) = N_s \dot{W}_s(t)$. The underlying assumption in the estimation algorithm is that the number of TCP flows and non-responsive flows does not change or changes very slowly.

Observable quantities of packet drop probability and queue length are sampled in each *sampling period* to estimate the desired fraction of non-responsive load, ψ :

$$\psi = 1 - \frac{z(t)}{D+z(t)}. \tag{11}$$

By taking a second derivative of q using eqn. 9, we get

$$\ddot{q}(t) = \frac{\dot{z}(t)}{R(t)} - \frac{z(t) + D}{R^2(t)} \dot{R}(t). \quad (12)$$

Combining eqn. 12 with eqn. 8, we get

$$R(t)\ddot{q}(t) + \left(\frac{\dot{q}(t)}{C} + 1\right)\dot{q}(t) = \frac{N_s}{R(t)} - \frac{z(t)z(t-R(t))}{2N_s R(t)} p(t-R(t)) \quad (13)$$

In congestion avoidance phase of TCP, $z(t)$ changes very slowly with time. So we are able to rewrite eqn. 13 according to the regression model [15] as follows,

$$R(t)\ddot{q}(t) + \left(\frac{\dot{q}(t)}{C} + 1\right)\dot{q}(t) = \begin{bmatrix} \frac{1}{R(t)} & -\frac{p(t-R(t))}{2R(t)} \end{bmatrix} \begin{bmatrix} N_s \\ \frac{z(t)z(t-R(t))}{N_s} \end{bmatrix} \quad (14)$$

In case that there is a infrequently sudden change or a slow change in traffic, the algorithm can be reset or applied with weighted average method respectively so that we still can apply regression model to the algorithm.

We base the estimation algorithm on “small signal” behavior and we have: $q(t) \approx q_0$; $R(t) \approx R_0 = \frac{q_0}{C} + T_p$; $z(t) \approx z_0 \approx z(t - R_0)$.

With approximations mentioned above, we define the following equations from eqn. 14:

$$\chi(t) = R(t)\ddot{q}(t) + \left(\frac{\dot{q}(t)}{C} + 1\right)\dot{q}(t) \quad (15)$$

$$= \mathbf{W}^T(t)\boldsymbol{\beta}^* \quad (16)$$

$$\chi_e(t) = \begin{bmatrix} \frac{1}{R(t)} & -\frac{p(t-R(t))}{2R(t)} \end{bmatrix} \begin{bmatrix} N_s \\ \frac{z(t)z(t-R(t))}{N_s} \end{bmatrix}$$

$$\approx \begin{bmatrix} \frac{1}{R_0} & -\frac{p(t-R_0)}{2R_0} \end{bmatrix} \begin{bmatrix} N_s \\ \frac{z_0^2}{N_s} \end{bmatrix}$$

$$= \mathbf{W}^T(t)\boldsymbol{\beta} \quad (17)$$

To obtain \dot{q} and \ddot{q} , we use a differentiation scheme. And they are given by: $\dot{q}(t) = (q(t) - q(t-1))/T$; $\ddot{q}(t) = (\dot{q}(t) - \dot{q}(t-1))/T$. T is a sampling interval.

In eqn. 15, all parameters can be easily obtained from the router or through simple calculation. So we refer eqn. 15 as measurement value or observed variable, to which we try to keep estimation value close. In eqn. 17, $\mathbf{W}^T(t)$ is a regression vector and known by measurement and simple calculation. But $\boldsymbol{\beta}$ is the unknown vector to be determined.

In the real-time parameter estimation, $\boldsymbol{\beta}(t)$ is the “best estimate” of $\boldsymbol{\beta}^*$ at time t . $\chi_e(t)$ is the predicted value of $\chi(t)$. The error $\mathbf{e}(t)$ between $\chi(t)$ and $\chi_e(t)$ can be used to update $\boldsymbol{\beta}$ recursively. The error $\mathbf{e}(t)$ is given by:

$$\mathbf{e}(t) = \chi(t) - \chi_e(t)$$

There are several recursive algorithms available in the real-time parameter estimation. One of those simplified algorithms is Kaczmarz’s projection algorithm [15]. It leads to less computation complexity and quick convergence. With modified Kaczmarz’s projection algorithm, we can fine-tune the algorithm via user-defined factors(α and γ). We apply the projection algorithm in the following manner:

$$\beta(t+1) = \beta(t) + e(t) \frac{\gamma \mathbf{W}(t)}{\alpha + \mathbf{W}^T(t) \mathbf{W}(t)}, \quad \text{where } \alpha \geq 0 \text{ and } 0 < \gamma < 2 \quad (18)$$

Once we get β_0 and β_1 from eqn. 18, we can calculate estimation values as follows:

$$\begin{aligned} N_s &= \beta_0 \\ z_0 &= \sqrt{\beta_0 \beta_1} \end{aligned}$$

Then we plug estimation results obtained from equations above into eqn. 11 and get the estimation of the fraction of incoming non-responsive load. It is noted that our algorithm measures this fraction relative to the arrival rate at the switch, and not relative to the capacity C of the outgoing link.

After initialization, the algorithm collects history information of $\mathbf{p}(t)$ and $\mathbf{q}(t)$. Then the algorithm utilizes history information of several data points before current one to compute the estimation with simple matrix operations, according to equations shown above. The current data point becomes history information of later computation. So the computation complexity is $O(1)$.

Note that the algorithm only depends on drop probability and queue length. Since the required number of history information of $\mathbf{q}(t)$ and $\mathbf{p}(t)$ is relatively small, the use of memory resource is also limited.

4 Implementation and Results

We implemented our estimation algorithm in RED module of Network Simulator (ns-2). Our algorithm depends on drop probability and the queue length. There are a number of options for measuring drop probability and queue length. We could use instantaneous measured values or values averaged over certain period or weighted averages that consider previously measured values. Instantaneous values could be quite noisy based on incoming traffic’s burstiness. Smoothed values try to reduce the impact of this noise by considering averages over longer periods. We considered five possible combinations of measuring the drop probability and the queue length: (ip, iq), (ip, aq), (ap, aq), (ap, iq) and (wp, wq) where prefix i stands for instantaneous, a stands for average and w stands for weighted average. Instantaneous queue length is the queue length at the time of sampling. Instantaneous drop probability is obtained by plugging the instantaneous queue length into the buffer management (RED) drop function employed at the router. Average drop probability is computed by dividing the number of dropped packets by the number of arrived packets during the sampling interval. Average queue length is computed over the sampling interval. Weighted drop probability and weighted queue lengths are computed by an exponential decay function similar that used in TCP i.e., $w\mathbf{p} = \alpha \times w\mathbf{p}_{old} + (1 - \alpha) \times \mathbf{ap}$ of current sample interval and $w\mathbf{q} = \alpha \times w\mathbf{q}_{old} + (1 - \alpha) \times \mathbf{aq}$ of current sample interval, where α is the forgetting factor.

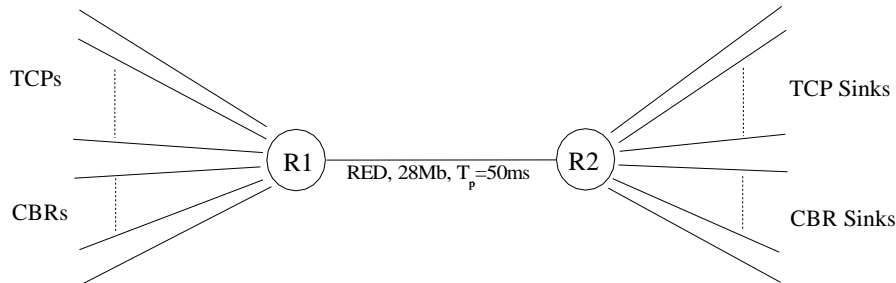


Figure 1: Simulation Topology

We used a bottleneck link topology shown in Fig. 1 for our simulations. We used an RED drop function with $(min_threshold, max_threshold, p_{max}) = (15, 45, 0.1)$ for managing the queue. The bottleneck router has 60 buffers and a link capacity of 28Mb and a propagation delay of 50ms. We used 1 Mbps CBR flows to model non-responsive flows. Long-term TCP flows are modeled by FTP flows. Packet sizes of 1000 bytes are used. A sampling interval of 17ms (corresponding to 1/3rd of the propagation delay at the link) is used.

Our algorithm computes an estimate every sampling interval. Several of these estimates are averaged to produce an estimate over a larger time interval called *estimate interval*. We chose 40 sampling intervals to equal one estimate interval. With our choice of parameters, this is roughly 700ms. Again to smooth variations, estimates are weighted-averaged over estimate intervals i.e., $new\ estimate = 0.8 \times previous\ estimate + 0.2 \times current\ estimate$.

Each estimation interval, we compute the actual fraction of non-responsive traffic by counting the packets of non-responsive flows and dividing it by the total number of arrived packets. The actual values are compared to the estimates. We use Mean Square Error(MSE) and Relative Error(RE) to describe the accuracy of the estimation compared to the measurement value. Mean square error is computed as $\sum_{i=1}^n (estimate_i - actual_i)^2 / n$ and relative error is computed as $\sum_{i=1}^n (estimate_i - actual_i) / \sum_{i=1}^n actual_i$, where n is the total number of estimations output by algorithm during the simulation.

4.1 Comparison of Different Ways to Collect Inputs

As explained earlier, there are many ways to compute the drop probability and the queue length at the router. We conducted experiments to see which one of these several combinations leads to better estimations. In this experiment, we compare the impact of different combinations of collecting inputs (drop probability and queue length) on the accuracy of estimation. We set up 35 responsive flows and 22 non-responsive CBR flows. The simulation results are shown in Table 1.

From Table 1, we notice that (wp,wq) has the minimal MSE and RE. In the rest of the simulations, we use (wp,wq) in our implementation of the algorithm. Since our estimation algorithm relies on \hat{q} and \hat{q} , it is necessary that we don't average the queue lengths over large periods of time. We used a

Table 1: Comparison of Combination of Inputs

Comb. of Inputs	MSE	RE
(wp,wq)	0.002547	-0.022
(ip,iq)	0.051447	-0.292
(ip,aq)	0.007430	0.046
(ap,iq)	0.031454	-0.228
(ap,aq)	0.007213	-0.087

forgetting factor α of 0.1 for queue lengths and α of 0.9 for drop probability.

4.2 High Non-responsive Load

In this experiment, we set up 35 responsive flows and 19, 22 and 25 non-responsive CBR flows respectively. Each CBR flow pumps packets at the rate of 1Mbps. The estimation and measurement values are shown in Figure 2 below.

We observe from the results that the estimation fluctuates around actual measured value in a very small range, providing a good estimate of the actual non-responsive load. This indicates that the estimation algorithm seems to work well at high non-responsive loads. It is noted again that the fraction of the load that is non-responsive is measured relative to the arrival rate at the switch and hence the measured fraction varies over time as the responsive traffic arrival rate changes over time.

4.3 Changing non-responsive Load

In order to study the impact of different levels of non-responsive load on the estimation algorithm, we conducted several experiments where we fixed the number of responsive flows at 35 and varied the number of non-responsive flows to vary the fraction of non-responsive traffic at the router. Simulation results are shown in Figure 3. MSE and RE are plotted against proportion of non-responsive flows in the workload. During these simulations, drop rates varied from 1.56% to 5.64%.

From Figure 3, we notice that as we increase the non-responsive load, the estimation is more accurate. The error within the range of (40-85%) non-responsive load is quite acceptable, within 15% of the actual value. One interesting observation is that at lower non-responsive loads, the estimation algorithm is less accurate than at higher non-responsive loads.

Our estimation method relies on queue dynamics as seen by the use of \hat{q} and \ddot{q} in the algorithm. Hence, our algorithm tends to be more accurate when these quantities can be accurately measured. When the arrival rate at the switch is low, the queue lengths tend to remain low. At these times, when queue lengths don't fluctuate widely, the input excitation for our algorithm tends to be low to provide an accurate estimation. As a result, our model tends to be more accurate in higher load situations.

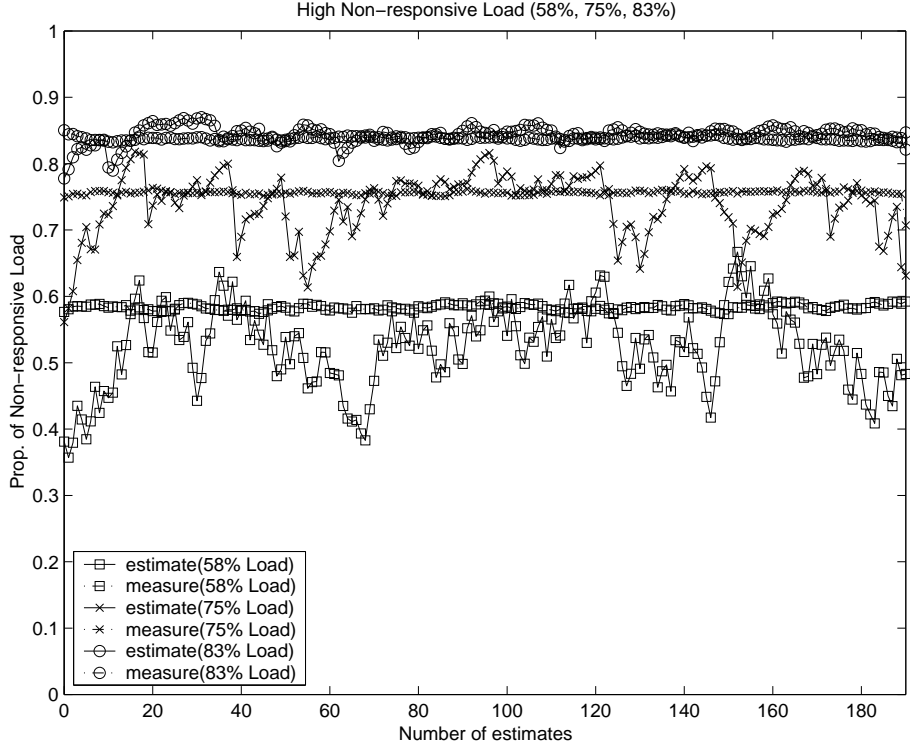
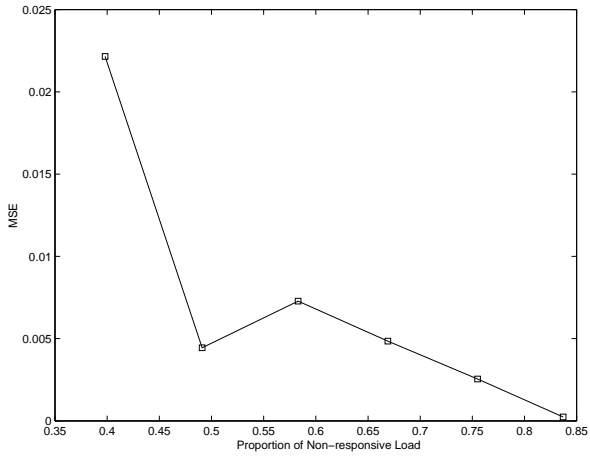
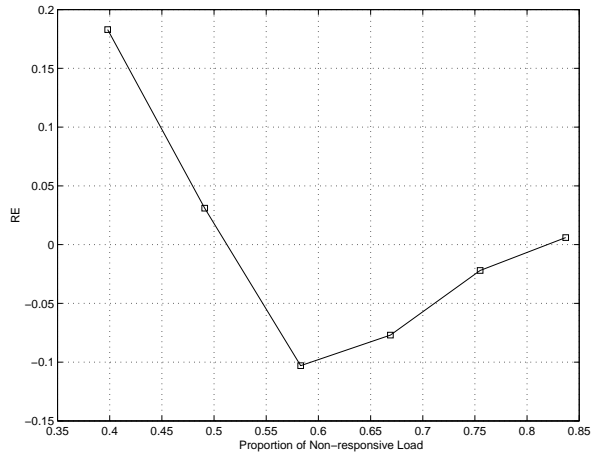


Figure 2: High Non-responsive Load



(a) MSE vs. Non-responsive Load



(b) RE vs. Non-responsive Load

Figure 3: Accuracy vs. Non-responsive Load

4.4 Dynamic Response to Traffic Change

To test the dynamic response of our algorithm to variations in the traffic, we conducted the following experiments. We initiate the simulator with a number of responsive and non-responsive flows. Of the 26 non-responsive flows, 6 of them are of ON/OFF type flows. They are ON for "x" number of seconds and OFF for the next "x" number of seconds. As a result, the non-responsive load has a square wave shape with a period of "2x". In our experiments, we use $x = 100$ seconds, 20 seconds, and 5 seconds. The results from these experiments are shown in Figure 4.

We observe from Figure 4 that the estimation algorithm can follow the dynamics of the input traffic fairly well. It is possible to make the algorithm follow quicker changes in traffic by averaging the estimates over smaller estimation intervals than what is used here in simulations, 700ms.

4.5 Mixed Traffic

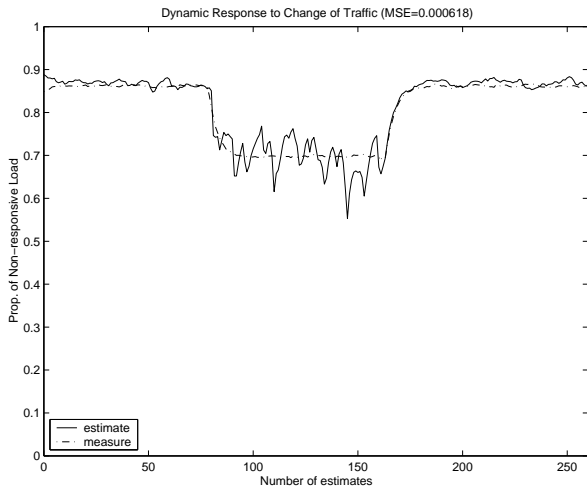
To simulate more realistic traffic environments, we simulated a mixed traffic load consisting of short-term TCP flows, long-term TCP flows and a number of non-responsive flows. Initially, the traffic consists of only long-term TCP flows and non-responsive flows. At 100s, we start 300 short-term TCP flows. Each short-term flow behaves like an ON/OFF flow, sending a certain number of packets at a random time and remains OFF for a certain time after that. Each short-term TCP flow repeats this procedure for 5 times in a 50 second time span. In the Figure 5, we show the results of our estimation algorithm when short-term TCP flows send 20pkts during each ON period. Notice that the estimation algorithm counts the proportion of short-term TCP load as part of non-responsive load. These flows do not persist in the network long enough to observe significant number of packet drops and hence appear to be non-responsive. Similar observations about short-term flows have been made in a number of recent studies [5, 16].

4.6 Impact of RTTs

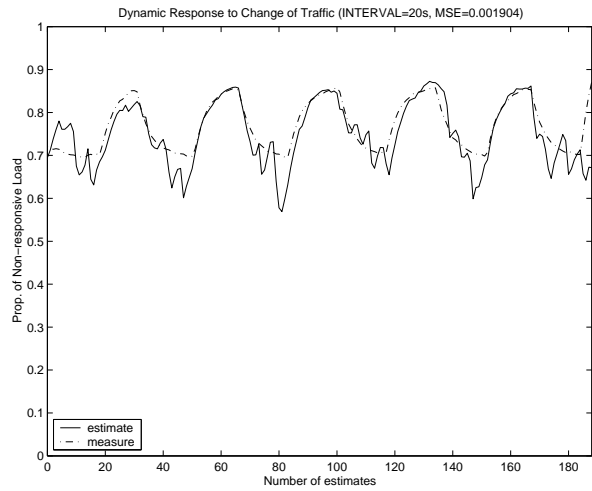
In order to study the impact of different RTTs on our algorithm, we conducted an experiment with 20 TCP flows with different RTTs, ranging from 30ms to 410ms. In the algorithm, we set the R_0 to the average value of the range. The result is shown in Figure 6.

We observe from Figure 6 that the estimation is still quite acceptable, although the algorithm over-estimates a little. Notice that we assume all TCP flows have the same RTT in the traffic model. However, the results here show that as long as we employ a reasonable average value in the estimation algorithm, the different RTTs of different flows do not impact the accuracy significantly. Recent studies based on wavelets provide a convenient way to estimate the range of RTTs of flows passing through a router [17].

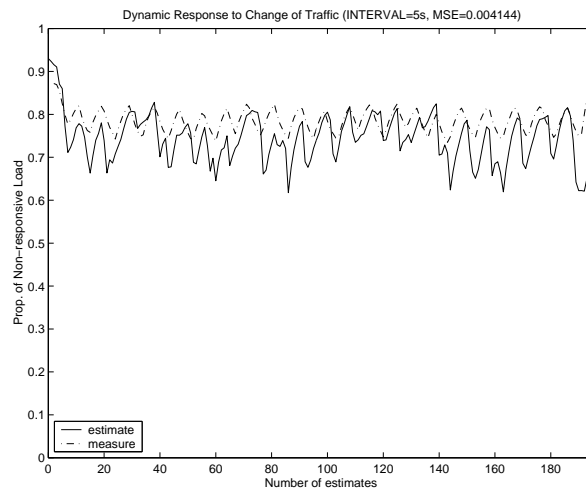
In order to further study the impact of RTT on the estimation, we set R_0 at the router to 100ms and run different experiments with different RTTs for flows. We fix the RTTs of all the flows at a single value, varying from 10ms to 200ms. The results are shown in Figure 7.



(a) One-time Change (100s)



(b) Continuous Change (Interval=20s)



(c) Continuous Change (Interval=5s)

Figure 4: Dynamic Response to Change of Traffic

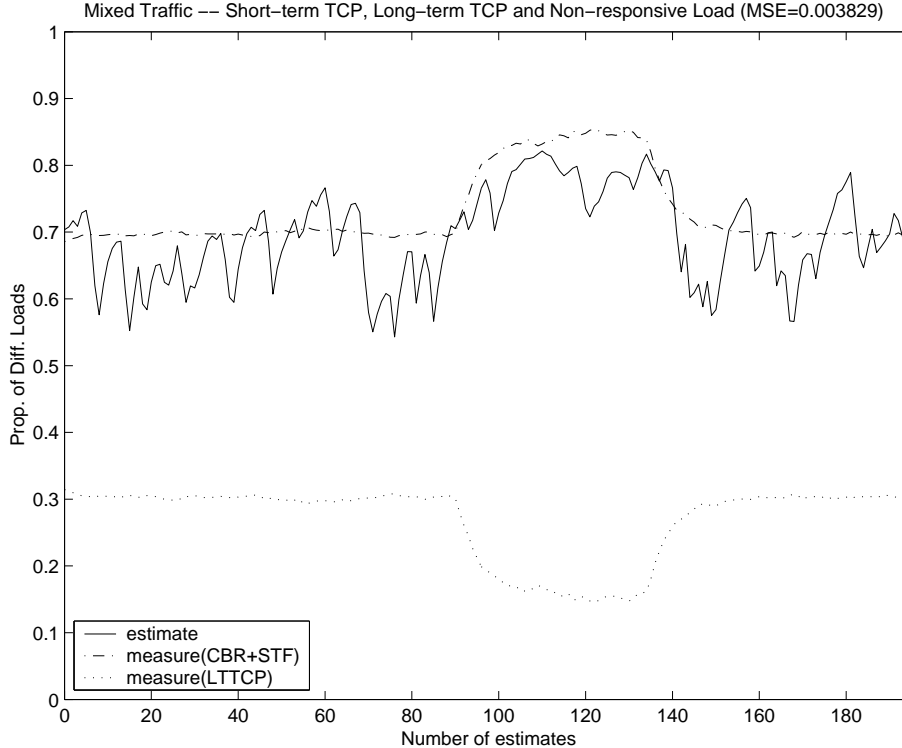


Figure 5: Traffic Mix – Short-term TCP, Long-term TCP and Non-responsive Load

From Figure 7, we notice that when the RTT of the flows is below R_0 , the algorithm overestimates the non-responsive traffic (since it underestimates the responsive TCP traffic). When the RTT of the flows is higher than R_0 , the algorithm underestimates the non-responsive traffic. However, it is observed the relative errors are within 20% even over such a wide range of RTTs.

5 Discussion and Future Work

With an aim to estimate the fraction of incoming traffic that is non-responsive, we have presented an estimation algorithm. Our estimation method relies on queue dynamics as seen by the use of $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$ in the algorithm. Hence, our algorithm tends to be more accurate when these quantities can be accurately measured. When the arrival rate at the switch is low, the queue lengths tend to remain low. At these times, when queue lengths don't fluctuate widely, the input excitation for our algorithm tends to be low to provide an accurate estimation. As a result, our model tends to be more accurate in higher load situations. It is at these times of higher loads that traffic engineering decisions or potential attack detections need to be made. Hence, our algorithm seems well suited for such situations.

Our work has focused on estimating the aggregate amount of non-responsive traffic at the router in contrast to the earlier suggestions for implementing checks to see if individual flows are responding to congestion [4]. Our work has relied on much of the earlier work on modeling traffic dynamics [9, 10, 11].

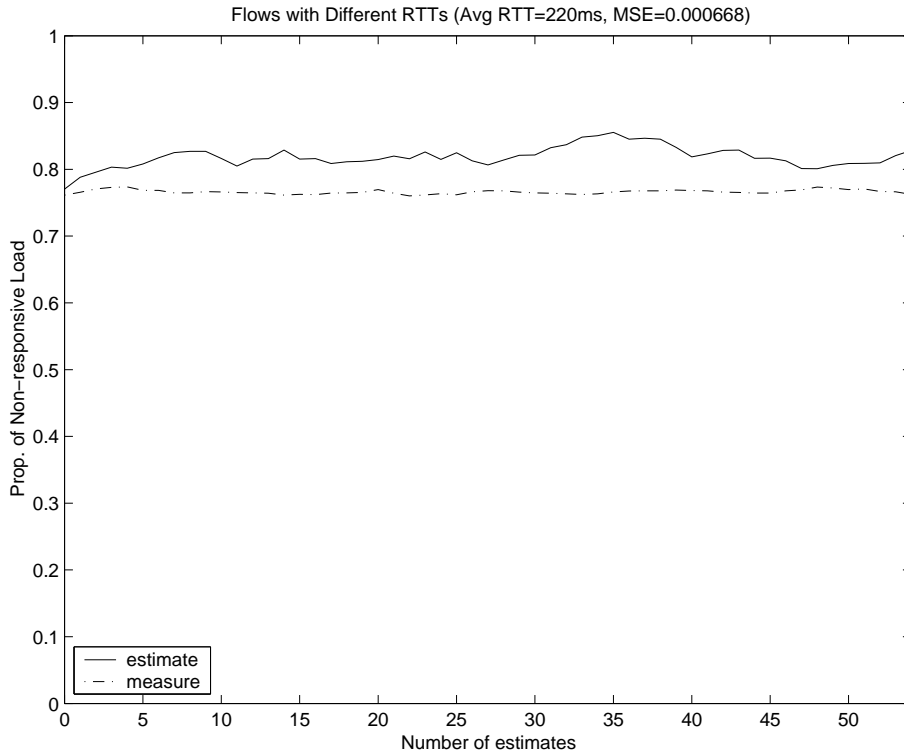
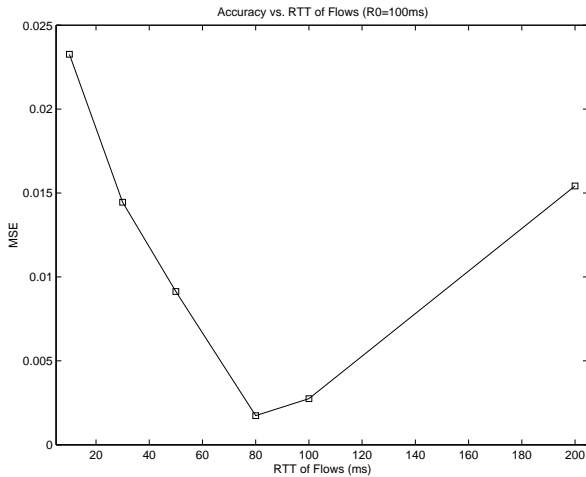
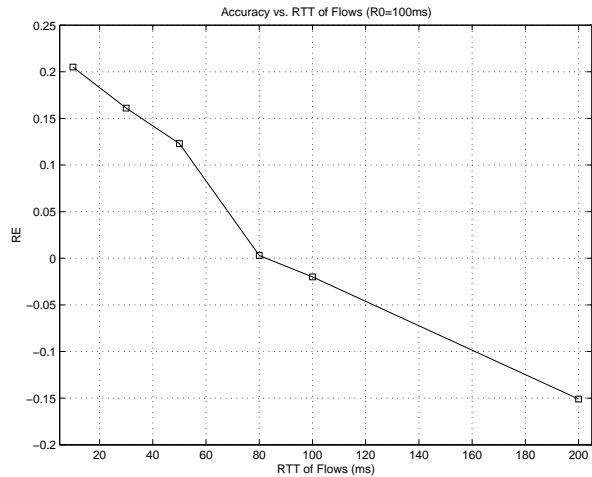


Figure 6: TCPs with Different RTTs



(a) MSE vs. RTT of Flows (R0=100ms)



(b) RE vs. RTT of Flows (R0=100ms)

Figure 7: Accuracy vs. RTT of Flows

Our work, as discussed in the introduction is motivated by traffic engineering concerns. We expect that the estimation of non-responsive traffic could lead to the following applications: (a) providing a handle on controlling non-responsive traffic; It is possible that an attack detection mechanism could be developed based on a robust estimation of non-responsive traffic. We could probably declare an "attack" if the fraction of non-responsive traffic exceeds certain acceptable threshold. If a robust detection mechanism could be developed based on the presented algorithm, this could provide an additional mechanism to counter attack traffic within the network. (b) provide a mechanism for tuning traffic control algorithms at the time of congestion. In [7], it is shown that in the presence of high non-responsive loads, drop tail buffer management may be better than RED style active queue management. Our estimation algorithm could possibly be used to drive such decisions at the times of congestion. (c) provide a better control for enforcing service differentiation. Earlier work on analyzing assured forwarding in differentiated services has shown that non-responsive traffic may disrupt service for responsive sources even when traffic is marked differently at the edge. With our estimation algorithm, it is possible to adapt the traffic control parameters to provide better service. These applications are subject of our future work.

Our initial results based on our estimation algorithm are promising. The general problem of building a robust estimation tool, however, is more challenging. In order to make the tool more robust, we are planning to do the following extensions: (a) incorporate a model of short-term TCP flows to differentiate them from non-responsive attack flows in order to provide a better attack detection mechanism, (b) use a more detailed model of TCP behavior that includes timeouts to extend the useful operating range of the estimation algorithm, and (c) identify techniques to systematically choose proper parameters in the algorithm for different applications.

6 Conclusion

In this paper, we have provided a model for mixed traffic at a router. The model was used in deriving an algorithm for estimating the fraction of the incoming traffic that is non-responsive to congestion. We have shown that the proposed algorithm works acceptably in a wide range of traffic scenarios through ns-2 based simulations. We observed that persistence of excitation plays a significant role in the accuracy of the proposed estimation algorithm. We are in the process of making our algorithm more robust based on these observations. Possible future applications of the proposed algorithm were discussed.

References

- [1] National Laboratory for Applied Network Research, "Network traffic packet header traces," <http://moat.nlar.net/Traces/>, 2000.
- [2] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-based congestion control for unicast applications," *Proc. of ACM SIGCOMM*, Aug. 2000.
- [3] D. Bansal and H. Balakrishnan, "TCP-friendly congestion control for real-time streaming applications," *Proc. of Infocom*, Apr. 2001.

- [4] S. Floyd and K. Fall, "Promoting the use of end-to-end congestion control in the internet," *ACM/IEEE Trans. on Networking*, Aug. 1999.
- [5] End to-end discussion group, "Web mice versus elephants," *Mailing list discussion*, 2000-2001.
- [6] M. Christiansen, K. Jeffay, D. Ott, and F. D. Smith, "Tuning RED for web traffic," *Proc. of SIGCOMM'00*, Aug. 2000.
- [7] Christophe Diot, Gianluca Iannaccone, and Martin May, "Aggregate Traffic Performance with Active Queue Management and Drop from Tail," *Sprint ATL Technical Report TR01-ATL-012501*, 2001.
- [8] Packeteer Inc., "Packetshaper: Opening a world of intelligent bandwidth management services," *Technology white paper*, <http://www.packeteer.com/>, 1998.
- [9] V. Misra, W. B. Gong, and D. Towsley, "A fluid based analysis of a network of AQM routers supporting TCP flows with an application to RED," *Proc. of ACM SIGCOMM*, Aug. 2000.
- [10] C. V. Hollot, V. Mishra, D. Towsley, and W. Gong, "A control theoretic analysis of RED," *Proc. of Infocom*, Apr. 2001.
- [11] S. Kuniyur and R. Srikant, "Analysis and design of an adaptive virtual queue algorithm for active queue management," *Proc. of ACM SIGCOMM'01*, Aug. 2001.
- [12] C. V. Hollot, V. Mishra, D. Towsley, and W. Gong, "On designing improved controllers for AQM routers supporting TCP flows," *Proc. of Infocom*, Apr. 2001.
- [13] University of California at Berkeley, "Network simulator (ns)," *Available via <http://www-nrg.ee.lbl.gov/ns/>*, 1997.
- [14] E. Altman, K. Avrachenkov, and C. Barakat, "Stochastic Model of TCP/IP with Stationary Random Losses," *Proc. of SIGCOMM'00*, Aug. 2000.
- [15] K. J. Åström and B. Wittenmark, *Adaptive Control*, Addison Wesley, Reading, MA, 1995.
- [16] N. Caldwell, S. Savage, and T. Anderson, "Modeling tcp latency," *Proc. of IEEE Infocom*, March. 2000.
- [17] P. Huang, A. Feldmann, and W. Willinger, "A non-intrusive, wavelet-based approach to detecting network performance problems," *Proc. of Internet Measurement Workshop*, Nov. 2001.