Quantifying Responsiveness of TCP Aggregates by Using Direct Sequence Spread Spectrum CDMA and Its Application in Congestion Control

Mehdi Kalantari
Department of Electrical and Computer Engineering
University of Maryland, College Park, MD 20742
Phone: 301 405 8841, Email: mehkalan@eng.umd.edu

Mark Shayman
Department of Electrical and Computer Engineering
University of Maryland, College Park, MD 20742
Phone: 301 405 3667, Email: shayman@eng.umd.edu

Abstract

In this paper, we introduce the notion of responsiveness for TCP aggregates and define it as the degree to which a TCP aggregate reduces its sending rate to the network as a response to packet drops. We define metrics that describe the responsiveness of TCP aggregates, and then we suggest two methods for determining the values of these quantities. The first method is based on a test in which we drop a few packets from the aggregate intentionally and measure the resulting rate decrease of that aggregate. This kind of test is not robust to multiple simultaneous tests performed at different routers. Extensions are done to make the test robust to multiple simultaneous tests by using ideas from the CDMA approach to multiple access channels in communication theory. Then we use these methods to perform congestion control. A distinguishing feature of our congestion control scheme is that it maintains a degree of fairness among different aggregates.

Keywords: Responsiveness of TCP Aggregates, CDMA, Aggregate Perturbation Method, and Congestion Control.

I. INTRODUCTION

A key characteristic of TCP traffic is its responsiveness to packet drops. This means that the TCP aggregates reduce their sending rate as a result of experiencing packet drops made by the network at the time of congestion. The degree to which a TCP aggregate reduces its rate in response to packet drops depends on packet size, round trip time and the distribution of window sizes among the constituent flows.

The goal of this paper is to introduce a technique for quantifying the responsiveness of a TCP aggregate to packet drops and applying the measure of responsiveness to obtain a fair congestion control scheme. In this scheme, when a router faces congestion and needs its traffic arrival rate to be reduced, it tries to drop the packets of less responsive aggregates more aggressively.

One of the unique features of our work is that we try to apply the responsiveness measure at the aggregate level, not at the flow level. In general an aggregate is a group of flows with a common property that pass through the same router or switching device at some point in the network. An example of an aggregate can be all FTP flows that pass through a router, or all the traffic being routed toward yahoo.com.

Performing the responsiveness test in the aggregate level has several advantages. First, it does not suffer from scalability; many flows can be bundled together to form an aggregate and the responsiveness test is done for the resulting aggregate. The second advantage of aggregate based testing is the fact that the majority of the current traffic of Internet is composed of short-lived flows known as Internet mice. It is extremely hard to perform responsiveness tests for such traffic in the flow level because flows last only for a few round trip times, and often they end before a router can keep track of them. However, if we put many such flows together, we will get an aggregate that is composed of many flows that appear, survive for a few round trip times and disappear. The aggregate composed of these flows has some statistical properties that can help us to define a responsiveness measure for it.

In general terms, our approach to measure the responsiveness of an aggregate is to perturb the arrival rate of the aggregate by intentionally dropping a small number of packets, and observing the way the aggregate responds. A normal TCP aggregate shows a transient degradation in its rate as a result of instantaneous packet drops, and we measure this degradation and use it as a responsiveness measure. By doing this periodically, the responsiveness of the aggregate can be determined. We have called this approach the Aggregate Perturbation Method (APM).

The above approach has a drawback in a distributed implementation. Each router should be able to apply perturbations and use these perturbations to determine the responsiveness of the aggregates it observes. However, the flows in an aggregate may experience perturbations at multiple routers. In a distributed implementation, in order to perform its test, a router should not need to be aware of the perturbations applied by other routers. Our approach to solving this problem is inspired by the direct sequence spread spectrum (CDMA) approach in multiple access communication channels. Each router is assigned a dropping signature that specifies its packet dropping rate as a function of time. Different routers are assigned signatures that are orthogonal in a certain sense. Using simulations, we show that this approach enables each individual router to find the response coefficient of the aggregates that pass through it without requiring any information to be shared among routers. We have called this approach CDMA based Aggregate Perturbation Method (CAPM).

As an application of APM and CAPM, we use the response coefficients to offer a fair congestion control mechanism. In this scheme the response coefficients are taken into account to assign the drop probability of different aggregates at the time of congestion; less responsive aggregates are penalized more by having a higher probability of drop. Therefore, different aggregates will show the same absolute value or percentage of rate decrease. Our simulation results confirm the effectiveness of this scheme to keep fairness among the aggregates.

The remainder of the paper is organized as follows. In Section II, we describe related work. In Section III, we explain APM. In Section IV, we introduce CAPM and describe how the use of CDMA-inspired orthogonal perturbing signatures enables multiple routers to perform perturbations without mutual interference. In Section V, we will explain our approach to using the response coefficients of aggregates for the purpose of congestion control. In Section VI, the results of the simulation experiments are presented which confirm the efficacy of the proposed methods.

II. RELATED WORK

Many researchers have conducted studies to do identification and modelling of TCP traffic in the granularity of flow under steady state conditions. In [1] the authors propose a method of testing a flow by comparing the steady state throughput of a TCP flow with the theoretical predicted value for conforming flows. If the response and model are similar, the flow is called TCP conforming. The objective of that study is to identify and penalize the nonconforming flows for congestion control purposes. The approach in [1] describes how large sustained individual flows may be tested for TCP conformance. However, a significant percent of the Internet traffic may be composed of short lived flows.

Stochastic Fair Blue (SFB) is proposed in [12], and it offers per flow test for responsiveness by mapping different flows to parallel bins. The approach is based on the fact that the bins containing a nonconforming flow are likely to be overloaded. However, if there are many nonconforming flows in a traffic aggregate, it is likely that all bins are overloaded, and the algorithm will not be able to distinguish between conforming and nonconforming flows.

There are many other works dealing with the TCP dynamics and its throughput analysis. In [11] the authors have offered the throughput model for a TCP traffic under assumption of stationary random losses. In [6] the authors offer a flow based analysis of TCP dynamics in the Active Queue Management (AQM) routers by using stochastic differential equations.

III. Aggregate Perturbation Method

In this section, we introduce the Aggregate Perturbation Method (APM) for quantifying the responsiveness of TCP aggregates. APM works based on instantaneously dropping a number of packets from an aggregate at some point and observing the resulting transient decrease in the rate of the aggregate.

For the purpose of this paper, we assume the TCP aggregates are composed of TCP flows that conform to TCP-Reno congestion control mechanism. TCP-Reno has two different phases known as *slow start* and *congestion avoidance*. Slow start begins after making a connection, and upon successful transmission of every packet and receiving acknowledgement from the receiver the window size is increased by one. Congestion avoidance starts after the window size exceeds a threshold value, and in this phase the window size is increased one per round trip time, and upon experiencing a drop it is decreased to half its current value.

Assume at some router we have an aggregate of TCP flows with arrival rate of $\lambda(t)$. In order to test the aggregate for responsiveness, at time $t = 0$, we drop D packets from it. It is expected that the aggregate responds to the packet drops by decreasing its rate for a while after time $t = 0$. We define the following responsiveness measure for the aggregate as a response to packet drops:

$$\eta(D) = \int_0^{t_r} (\lambda(0^-) - \lambda(t))\,\mathrm{dt} \tag{1}$$

in which $\lambda(0^-)$ is the instantaneous rate at the moment before dropping the first packet, and t_r is a nonnegative finite time, and it can be chosen to be the minimum time for the recovery of all flows that received drops (in the order of a few times the longest round trip in the aggregate). To achieve better results, $\lambda(0^-)$ may be replaced by a short-term average of the rate of the aggregate in a time interval earlier than $t = 0$. $\eta(D)$ is simply a measure of how many more packets could have been sent by the aggregate if we had not dropped D packets. This measure is illustrated in figure 1.

In [8] we have shown that the expected value of $\eta(D)$ is a linear function of D. Furthermore, this quantity does not depend on the number of flows contributing to the aggregate and the absolute value of the aggregate rate $\lambda(t)$ if the number of dropped packets D is small compared to the number of active flows at time $t = 0$.

Our approach for quantifying the responsiveness of a TCP aggregate is based on the degradation measure $\eta(D)$ as a response to packet drops; under the same value of D for different aggregates, those with a higher $\eta(D)$ are more responsive. In other words, $\eta(D)/D$ can give a quantitative value of the responsiveness of an aggregate.

IV. CDMA Based Aggregate Perturbation Method (CAPM)

One of the problems of distributed implementation for APM is the potential of simultaneous perturbations; the measurements of a perturbing router on an aggregate can be falsified by the simultaneous perturbations being done on the same aggregate in a downstream or upstream router. This phenomenon is illustrated in the figure 2. As it can be seen in this figure, the response of the APM test of a router at $t = t_1$ is overlapped by the response of the aggregate to another router's test at time $t = t_2$, which causes interference. This interference happens when t_1 and t_2 are close enough to each other (more precisely $|t_2 - t_1| < t_r$). In this case the measure given by equation (1) does not give accurate information about responsiveness of the aggregate, and interference causes the results of both tests to be falsified.

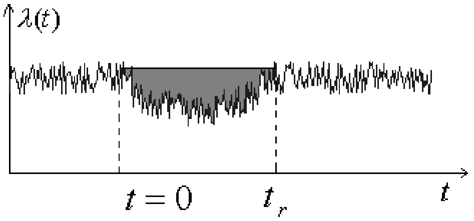

Fig. 1. The shaded area shows $\eta(D)$, the responsiveness measure defined by equation (1). D packet are dropped from the aggregate at $t = 0$.

In this section we introduce CAPM to overcome the above problem. In CAPM every perturbing router uses a unique perturbing pattern. We will show that under proper assignment of the perturbing patterns and proper definition of aggregate degradation measure for each perturbing router, the test and measurement of each router will be robust to the interference caused by the other simultaneous perturbing routers.

CAPM is different from APM in two ways. The first difference is that we spread the packet drops over time. In other words, instead of dropping D packets from the aggregate instantaneously at time $t = 0$, we spread the packet drops over a time interval $[0, T]$. In this scheme perturbation is done according to the packet drop rate function $r_i(t) : [0, T] \to \mathcal{R}$ for the i^{th} router. The responsiveness test and measurement is done during the interval $[0, T]$, and at time $t \leq T$, the i^{th} router drops $r_i(t)$ packets per second from the aggregate. We refer to $r_i(t)$ function as the *drop signature* of the i^{th} router.

The second difference between CAPM and APM is the way we define the degradation measure for the i^{th} router as the response to dropping with rate $r_i(t)$. In this case instead of the simple integral given by equation (1), we use a weighted integral to measure responsiveness of the aggregate under perturbation:

$$\eta_h(r_i) = \int_0^{T} h(t)\Delta\lambda(t)\,\mathrm{dt} \tag{2}$$

in which $\Delta\lambda(t) = \lambda(0^-) - \lambda(t)$, and $h(t)$ is a weighting function that states at what time instants the results are more important to us, and at what time instants we are less interested in the rate decrease of aggregate.

In the next step we try to use an approach similar to Direct Sequence Spread Spectrum CDMA in multiple access communication to solve interfering problems of multiple simultaneous perturbing routers [7]. In this approach, each router perturbs the traffic according to its unique drop signature based on a CDMA code assigned to it. The idea is that if we define the drop signature of different routers in a way that they are

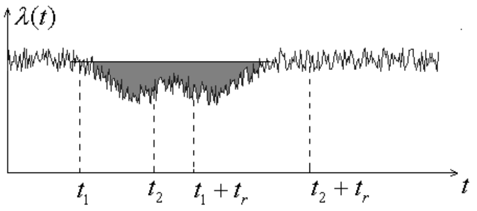

Fig. 2. The interference effect of simultaneous APM tests done by different routers. Before the aggregate recovers from a router's perturbation at $t = t_1$, another router performs a test at $t_2 < t_1 + t_r$. The results of both tests are falsified.

orthogonal to each other in a certain sense, then by proper definition of the weight function $h(t)$ the measure of degradation in a router defined in equation (2) will be independent of the perturbations done by the other routers.

Similar to the CDMA systems, we define the drop signature of the i^{th} perturbing router in the following way:

$$r_i(t) = A_i \sum_{j=1}^{N} c_j p_{T_c}(t-(j-1)T_c) = A_i s_i(t) \qquad (3)$$

in which A_i is a known perturbation amplitude of the i^{th} router, N is a positive integer called the spreading factor, $T_c = T/N$, $(c_1, c_2, \ldots, c_N)$ is a binary sequence assigned to the particular router known as the code of the router. In (3), $s_i(t)$ denotes the *normalized drop signature,* and $p_{T_c}(t)$ is a real-valued function known as the chip waveform and it satisfies the following property:

$$\int_{-\infty}^{\infty} p_{T_c}(t) p_{T_c}(t-nT_c)\,\mathrm{dt} = 0, \ \ \mathrm{n} = 1, 2, \ldots . \qquad (4)$$

The measurement of the i^{th} router about the responsiveness of the aggregate is made based on the *Matched Filter* output. The matched filter output is the value of $\eta_h(r_i)$ evaluated at $h(t) = s_i(t)$:

$$y_i = \int_0^T s_i(t)\Delta\lambda(t)\,\mathrm{dt}. \qquad (5)$$

Since in our problem $r_i(t)$ is a drop rate, it should be nonnegative, and hence $p_{T_c}(t)$ should be nonnegative. For this purpose we suggest the popular simple rectangular chip waveform:

$$p_{T_c}(t) = \left\{ \begin{array}{ll} 1 & if \ \ 0 < t < T_c \\ 0 & otherwise. \end{array} \right. \qquad (6)$$

Usually, in the CDMA systems assignment of the codes is very important. Users with a potential of high interference (e.g., neighbor routers in our problem) are assigned to codes that cause their drop signatures to be orthogonal (or close to orthogonal)

$$\int_0^T s_i(t) s_j(t)\,\mathrm{dt} = 0, \ \ \mathrm{for} \ \ \mathrm{i} \neq \mathrm{j}. \qquad (7)$$

Unfortunately, the statement of (7) cannot be satisfied with the current definition of drop signatures defined in (3). That is because both $s_i(t)$ and $s_j(t)$ are nonnegative rate functions, and hence the integral defined in (7) can never be zero. We can solve this problem by making a minor change of the orthogonality requirement and the structure of the matched filter. First, we replace the orthogonality condition by a similar condition in which the normalized drop signatures are orthogonal after removing their DC components:

$$\int_0^T s_i^a(t) s_j^a(t)\,\mathrm{dt} = 0, \ \ \mathrm{for} \ \ \mathrm{i} \neq \mathrm{j} \qquad (8)$$

in which $x^a(t)$ is $x(t)$ after eliminating its DC component over $[0, T]$:

$$x^a(t) = x(t) - \frac{1}{T}\int_0^T x(t)\,\mathrm{dt} \qquad (9)$$

Furthermore, we change the matched filter output for the i^{th} router in the following way:

$$y_i = \eta_{s_i^a}(r) = \int_0^T s_i^a(t)\Delta\lambda(t)\,\mathrm{dt} \qquad (10)$$

y_i is the value of η_h in (2) evaluated for $h(t) = s_i^a(t)$. One important fact about notation $\eta_h(r)$ in (10) is that in this equation r is the total perturbing function, since the rate decrease $\lambda(0^-) - \lambda(t)$ is affected by this total drop rate (i.e., $r(t) = \sum_k r_k(t)$, where k is an index that covers the set of all router perturbations that the aggregate experiences). It can be shown that if the total drop rate $r(t)$ is small enough compared to the rate of aggregate, then the system with input $r(t)$ and output the expected value of rate degradation $E[\Delta\lambda(t)]$ can be approximated by a linear system. In other words, the system can be linearized around its operating point.

Now we can state the following lemma; for the purpose of this lemma we assume $r_k(t)$ are piecewise constant functions as it was defined in (3).

Lemma 1: Assume that the overall drop rate $r(t) = \sum_k r_k(t)$ is small enough such that the system with input $r(t)$ and output $E[\Delta\lambda(t)]$ can be approximated by a linear system. Furthermore assume the holding time of the piecewise constant functions $r_k(t)$ on each constant interval is large enough compared to the response time of the aggregate. Then under the orthogonality assumption of (8) we have:

$$E[y_i] = E[\eta_{s_i^a}(r)] = E[\eta_{s_i^a}(r_i)] \qquad (11)$$

The proof of this lemma is given in the appendix. Note that the middle term of equation (11) is the measure of degradation with the weight function $h(t) = s_i^a(t)$ when all routers perturb the aggregate, however, the right term is the measure of degradation with the same weight function when only the i^{th} router perturbs the aggregate. The significance of Lemma 1 is that it states under orthogonality condition of equation (8) that the expected degradation measure at router i, $E[\eta_{s_i^a}(r)]$, is independent of perturbations being done at the other routers.

In Lemma 1 we have assumed that the holding time of $r_k(t)$ on the intervals on which it is constant is large enough compared to the aggregate response time. Generally, the response time of an aggregate is characterized by the round trip time of the flows contributing to it. Therefore for the piecewise constant function $r_k(t)$, the length of each constant interval should be significantly larger than the typical round trip time of the flows in the aggregate. This condition can be satisfied by making T_c long enough (e.g., 10 to 20 times the typical round trip time).

One useful observation about (10) is:

$$\int_0^T s_i^a(t)\lambda(0^-)\,\mathrm{dt} = 0 \tag{12}$$

And so we have the following simple equation for the output of the matched filter for the i^{th} router:

$$y_i = -\int_0^T s_i^a(t)\lambda(t)\,\mathrm{dt} \tag{13}$$

From (2) and (11), we have the following expression for the average output of the matched filter of the i^{th} perturbing router:

$$E[y_i] = E[\eta_{s_i^a(r_i)}]$$

This equation gives the basis for quantifying the responsiveness of TCP aggregates. Denote:

$$K_i = E[y_i]/A_i \tag{14}$$

Notice that K_i is a coefficient that describes how much the aggregate is responsive to packet drops. We call this quantity the *response coefficient* of the aggregate. Note that y_i is fully observable, and it can easily be measured by using (13). The amplitude of perturbing function A_i is known to the router that does the perturbation. Finding K_i is the only problem of the estimator. This coefficient can be estimated during the times that there is no congestion in the network. Or it can be estimated by a long term average of y_i/A_i based on multiple tests. Based on the result of Lemma 1, the estimation value of K_i is not affected by the perturbations done by the other routers, under the orthogonality assumption.

There are some key issues about how to choose the value of T_c. As stated before, T_c should be long enough such that the rate decrease of the aggregate as a result of packet drops in one chip duration can show up, and the aggregate rate settles down. On the other hand, too large T_c does not improve the performance in estimating the response coefficients, and it only causes longer test and more packet drops, which causes the test to be more expensive.

V. Fair Congestion Control by Using CAPM

In this section we suggest a method to use CAPM to do congestion control in a fair way. Random Early Drop [2] is one of the popular approaches to proactively prevent congestion in a router. By utilizing CAPM a router collects information about how responsive different aggregates are -i.e., K_i coefficients defined in the previous section. Knowing these coefficients helps a router to determine how much it should drop from each aggregate to reduce its bandwidth to a certain value.

Assume a traffic composed of many aggregates is intended to be forwarded through an outgoing link that has bandwidth shortage. So it is desired to keep the traffic bandwidth within the outgoing link capacity. If the router applies equal drop probability governed by a congestion controller such as a RED controller for all aggregates, the aggregates with higher response coefficients will back off more aggressively compared to the aggregates with smaller response coefficients. A certain degree of fairness among aggregates can be achieved by taking into account their response coefficients. Assume the traffic is a combination of M aggregates, and let $\lambda_i(t)$, and K_i denote the estimated instantaneous arrival rate and the response coefficient of the i^{th} aggregate respectively. Assume that we want to rate limit the total traffic, and let the output of congestion controller at time t be $p(t)$. With the information of response coefficients of aggregates the router can estimate how this total drop probability should be assigned among the aggregates to get a specific amount of rate decrease for each one.

To illustrate the above approach assume it is desired to have the same amount of rate decrease for all aggregates. Then we can assign the packet drops among different aggregates in a way that the product of the response coefficient and the drop rate is equal for all of them. In other words:

$$K_i\theta_i(t) = K_j\theta_j(t) \quad 1 \le i, j \le M \tag{15}$$

in which $\theta_i(t)$ and $\theta_j(t)$ denote the average drop rate of the i^{th} and j^{th} aggregate respectively. In the above equation subscripts are index of aggregates in the same routers. Heuristically equation (15) means that the rate decrease of the aggregates should be equal. It is important to note that equation (14) suggests using (15) as a heuristic to equalize the rate decreases of the aggregates; however, (15) is not a mathematical consequence of (14).

If $p_i(t)$ is the drop probability of the i^{th} aggregate, we have $\theta_i(t) = \lambda_i(t)p_i(t)$. Therefore, equation (15) can be written in the following way:

$$K_i\lambda_i(t)p_i(t) = K_j\lambda_j(t)p_j(t) \quad 1 \le i, j \le M \tag{16}$$

which gives $M-1$ linear equations. To find the numerical values of the drop probabilities we need one other equation. We use the fact that the total drop probability of the traffic should be $p(t)$. In other words:

$$\sum_{i=1}^{M} \frac{p_i(t)\lambda_i(t)}{\lambda(t)} = p(t) \tag{17}$$

in which $\lambda(t) = \lambda_1(t) + \lambda_2(t) = \ldots + \lambda_M(t)$ is the total rate of the traffic.

In the above approach we have tried to get the same rate decrease for different aggregates, however, one can apply the response coefficients in different ways to achieve an arbitrary value of rate decrease for each aggregate. For example, it may be desired to have the same percentage of rate decrease for different aggregates; in this case it is very easy to write equations similar to equation (16) to find drop probabilities.

VI. Simulations and Results

We have used the popular network simulator *ns2* to perform our experiments [10]. As explained previously, our focus is on TCP aggregates. For simulation we have used a network with fixed topology as in figure 3. The nodes $S_1, S_2, \ldots, S_n$ are n sources of TCP traffic. The propagation delay of the link between each source and router R_1 in figure 3 is different from

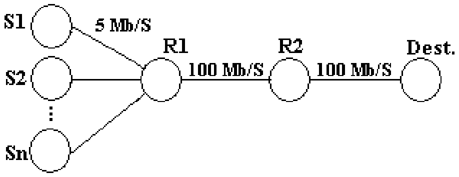

Fig. 3. The network topology used for simulation

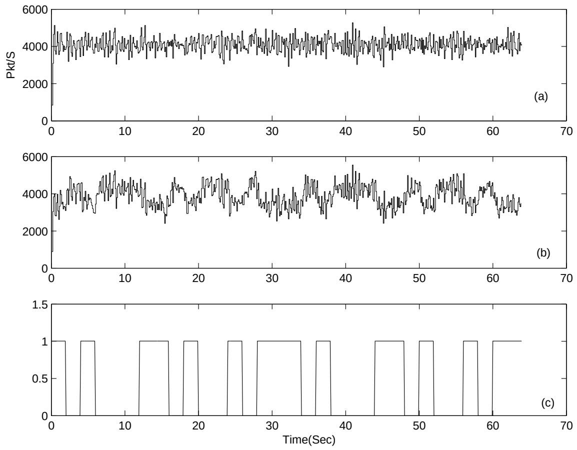

Fig. 4. a: The aggregate rate without any perturbation, b: The aggregate rate with perturbation of R_1, and c: the normalized drop signature of R_1.

a source to another source, and it has been chosen such that the round trip time of packets is uniformly distributed between 50 and 100 milliseconds under low congestion conditions. The flows at the sources are generated according to a birth-death process. Each source starts a TCP flow, and that flow ends after a random time uniformly distributed between 0 and 0.15 seconds. That source starts a new flow after waiting another random time uniformly distributed between 0 and 0.3 seconds. The packet size is constant equal to 1 Kbyte for all flows. In this topology, the link between R_1 and R_2, and also the link between R_2 and destination are bottleneck links. The capacity of these bottleneck links is 100 Mbps, that translates to 12500 packets per second.

In the first experiment we show how an aggregate responds to the signature based perturbations. For this experiment we run the simulation for two cases. In the first case the aggregate does not experience any perturbation; in the second case only R_1 perturbs the aggregate by using drop rate $r_1(t) = A_1 s_1(t)$, in which $s_1(t)$ is the normalized drop signature generated by plugging code $(1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1)$ in equation (3), and $A_1 = 160$ packet drops/Sec. In the simulation, the number of sources is 50, $T = 32$ seconds, $N = 16$, $T_c = 2$ seconds. Figure 4-(a) shows the rate of the aggregate when no perturbation is performed. In figure 4-(b) the rate of aggregate is shown when R_1 perturbs the aggregate by using $r_1(t)$, and figure 4-(c) shows two periods of the normalized drop signature $s_1(t)$. By inspecting figure 4-(b) we can see that the shape of drop signature of R_1 has appeared in the rate of the aggregate – with 180 degrees of phase shift. In other words, when $s_1(t) = 1$ (e.g., around $t = 14$), the rate of aggregate decreases, and when $s_1(t) = 0$ is 0 (e.g., around t=10), the rate increases.

In the second experiment we explore the typical response of aggregate when two routers perturb it simultaneously. In this experiment R_1 and R_2 perturb the aggregate by using different CDMA drop signatures. In the simulation, the number of sources is 50, $T = 32$ seconds, $N = 16$, $T_c = 2$ seconds. The code of R_1 is $(1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1)$, and that for R_2 is $(0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1)$. Under this assignment $s_1^a(t)$ and $s_2^a(t)$ are orthogonal. Two periods of the resulting normalized drop signatures for R_1 and R_2 are shown in figure 5-(b) and 5-(c) respectively. The amplitude of drop signatures for the two routers, A_1 and A_2, are the same and equal to 120 drops per second. Figure 5-(a) shows the rate of aggregate when the two routers R_1 and R_2 perturb the aggregate simultaneously. It can be seen that the additive shape of the two drop signatures appears on top of the aggregate rate– with 180 degree phase shift again. In other words, the two drop signatures modulate the aggregate rate additively. For example, at around time $t = 40$, the amplitude of both drop signatures is zero, and this shows up as an increase in the rate of aggregate as it can be seen in 5-(a) at $t = 40$. On the other hand, at time $t = 15$ or $t = 31$, the amplitude of both drop signatures is nonzero, and this shows up as a decrease in the rate at these two times.

The purpose of next experiment is to verify that under orthogonality definition of (8), the matched filter output of a router defined by (13) is not affected by perturbations done by the other routers. We proved this fact in Lemma 1. In this case we use the same CDMA drop signatures as in the previous experiment, but we change A_1 and A_2, the amplitude of the drop signatures of the two routers. Figure 6-(a) shows y_1, the output of matched filter for R_1 as it is defined by equation (13), when A_1 changes from 0 to 160 drops per second. In this figure, each $+$ represents a test in which R_1 perturbs the aggregate with drop signature $r_1(t) = A_1 s_1(t)$ and at the same time R_2 is also perturbing traffic with drop signature $r_2 = A_2 s_2(t)$, and $A_1 = A_2$. For each value of A_1 several tests have been done, and the average over multiple tests has been plotted by the solid line. It can be seen that the deviation of y_1 for each individual test from the average value shown by solid line is relatively small; this means that the matched filter output shows a small variance. The other observation about 6-(a) is linearity in amplitude of drop signature A_1.

In the other part of this experiment we turn off the perturbations done by R_2 by setting $A_2 = 0$, and do the same multiple test and measurement of y_1 for each value of A_1. The dashed line in figure 6-(a) shows the average of multiple tests for each value of A_1 for this case. It can be seen that the dashed line is very close to the solid line showing that perturbations of R_2 do not affect the output of matched filter of R_1. Figure 6-(b) is the same as figure 6-(a) for the second router.

In the next experiment we will show how the response coefficients can be used to do congestion control in a fair way. So we define two aggregates that pass through R_1. In this experiment the sources in the simulation network are divided into two groups. The on time of a flow generated by a source

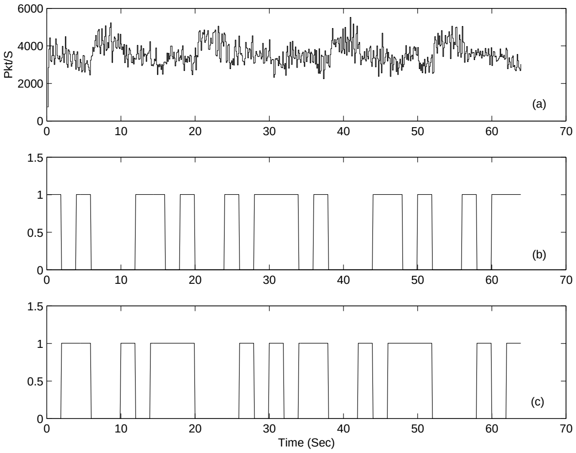

Fig. 5. a: The aggregate rate under two simultaneous perturbations, b: the normalized drop signature of R1, and c: the normalized drop signature of R2.

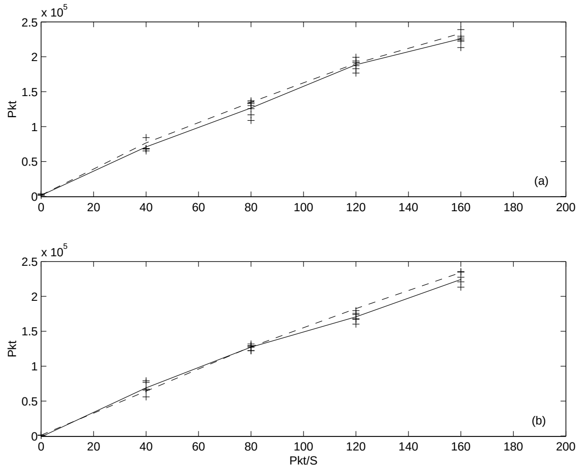

Fig. 6. Matched filter output versus the amplitude of drop signature, a: for the first router, b: for the second router.

in group 1 is uniformly distributed in $[0.15, 0.3]$ seconds, and after ending a flow, the source starts another flow after being idle for a random time uniformly distributed in $[0, 0.3]$. There are 50 sources in group 1. Sources in group 2 generate larger flows. The on time of a flow in group 2 is uniformly distributed in $[0.45, 0.9]$ seconds, and the idle time between flows is uniformly distributed in $[0, 0.5]$ seconds. There are 20 sources in group 2. We define the traffic generated by group 1 as the aggregate 1 and traffic generated by group 2 by the aggregate 2.

First we find the response coefficient of each of the two aggregates by using equation (14). The experiment shows that $K_1 = 77.3$, and $K_2 = 588.1$. The value of response coefficients have been found by several tests and averaging the results. The higher value of the response coefficient of aggregate 2 is easy to explain by considering the fact that the flows belonging to this aggregate are larger, so they show a higher rate decrease when they experience packet drops. The experiment shows that λ_1, the average rate of aggregate 1, is about 5234 pkt/sec, and $\lambda_2 = 4547$ pkt/sec. The total rate of the traffic is 9781 pkt/sec.

In the next step of this experiment, we doubled the number of sources in each group, so aggregate 1 needs about $2 \times 5234 = 10468$ pkt/sec; aggregate 2 needs 9094 pkt/sec, and the total demand is 19562 pkt/sec. This total demand is more than the link capacity which is 12500 packet/sec. The simulation results show that under drop tail condition in the forwarding queue of R_1, about 4% of incoming packets are dropped, and as a result of it the arrival rate of the traffic is reduced to about 12570 packets/sec. Under this drop policy the rate of aggregate 1 reduces to 8012 pkt/sec and the rate of aggregate 2 reduces to 4558 pkt/sec. The above data means that the rate reduction of aggregate 1 is 2456 pkt/sec or 25% of its demand, while the rate decrease of aggregate 2 is 4536 pkt/sec that is 52% of its demand. As it can be seen, aggregate 2 shows much higher rate decrease as a result of having a higher response coefficient.

To keep the fairness in rate reduction between the two aggregates, we use the fairness scheme explained in Section V to assign drop probabilities. By using equations (16) and (17) we find the drop probability $p_1 = 0.067$ for aggregate 1, and $p_2 = 0.010$ for aggregate 2, so the total drop rate of the traffic is still 4%. With these drop probabilities, the rate of aggregate 1 reduces to 6522 pkt/s, and rate of aggregate 2 reduces to 5523 pkt/sec. In this case the rate reduction of aggregate 1 is 3946 pkt/sec or 39% of its demand, and the rate reduction of aggregate 2 is 3571 pkt/sec or 42% of its demand. It can be seen that the rate reductions are much closer to each other than the previous experiment, and we were able to do a fair congestion control.

In the last experiment we will study how congestion can affect the measurement of response coefficients. For this purpose we used an aggregate like aggregate 1 with the same conditions that were stated in the previous experiment. The response coefficient of this aggregate was measured in independent simulation runs with different link utilizations of the bottleneck links. To increase the link utilization we increased the number of sending sources in group 1. The results have been shown in figure 7. In this figure each $+$ shows the response coefficient versus the link utilization. The response coefficient shows an almost flat behavior with reasonable variance up to the point where the link utilization is about 90%. After that the measurement of the response coefficient is not accurate, however, the measurements are still good approximations up to the point where the link utilization is about 95%. In this figure the solid line shows the average of the response coefficient over the experiments for which the link utilization is less than 90%; this average is about 75.

The degradation of performance of APM and CAPM in the presence of severe congestion is easy to explain; heavy congestion causes the aggregates to experience high rate of drops and as a result of that the aggregates shrink their rates. This causes them to become less responsive to the packet drops made by APM or CAPM. Although long term congestion is one factor that may degrade the performance of APM or CAPM, the APM and CAPM show a good performance in a wide range of link utilization before very heavy congestion happens. This can be one of the strong points about APM

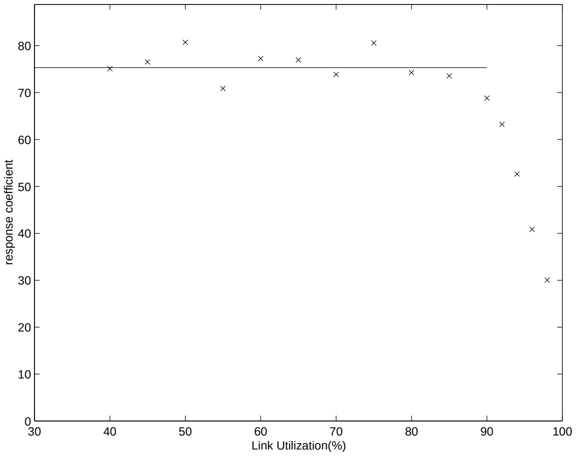

Fig. 7. The value of response coefficient of aggregate 1 measured at different link utilizations

and CAPM since these methods can be applied proactively to prevent congestion.

VII. Conclusions

In this paper, we introduced the Aggregate Perturbation Method (APM), and CDMA-based APM (CAPM), two techniques for quantifying the responsiveness of a TCP aggregate. Both algorithms perform a test on an aggregate by dropping some packets from it and observing the result. APM is the a simpler test but it is not robust to simultaneous tests at different routers. So we introduced CAPM that uses some unique drop signature for each router to do the test, and the approach is similar to the Direct Sequence Spread Spectrum CDMA in communication theory. We also defined a value called response coefficient to measure responsiveness of TCP aggregates to packet drops. We used these values for the purpose of fair congestion control among the aggregates with different response coefficients.

One important advantage of APM and CAPM is that they can be implemented in a distributed manner without needing data exchange between routers, and furthermore, these methods do not need any change in the current protocols. This also permits incremental deployment. One of the strong points about APM and CAPM is that these algorithms can perform proactively, and prevent congestion in an early stage.

References

[1] Floyd, S. and Fall, K., “Promoting the Use of End-to-End Congestion Control in the Internet,” *IEEE/ACM Transactions on Networking*, Vol. 7, No. 4, Aug 1999.

[2] Floyd, S. and Jacobson, V., “Random Early Detection Gateways for Congestion Avoidance,” *IEEE/ACM Transactions on Networking*, Vol. 1, No. 4, pp 397-413, Aug 1993.

[3] Jacobson, V., “Congestion Avoidance and Control,” *Proceeding of SIGCOMM 88*, Proceeding of SIGCOMM 88, Aug. 1988.

[4] Schulzrinne, H., “Long Term Internet Traffic Statistics,” available online at: http://www.cs.columbia.edu/ hgs/internet/traffic.html/.

[5] Savage, S.; Wetherall, D.; Karlin, A. and Anderson, T., “Network Support for IP Trace back,” IEEE/ACM Trans. on Networking, June 2001,Vol. 9, No 3.

[6] Misra, V.; Gong, W. and Towsley, D., “A fluid based analysis of a network of AQM routers supporting TCP flows with an application to RED ,” Proc. of SIGCOMM 2000, August 2000.

[7] Verdu, S. “Multiuser Detection ,” Cambridge University Press, 1998.

[8] Kalantari, M; Gallicchio, K. and Shayman, M., “Using Transient Behavior of TCP in Mitigation of Distributed Denial of Service Attacks,” Proc. of IEEE Conference on Decision and Control, Las Vegas, Nevada, December 2002.

[9] Mahajan, R.; Bellovin, S.; Floyd, S.; Ioannidis, J.; Paxson, V. and Shenker S., “Controlling High Bandwidth Aggregates in the Network ,” available online at: http://www.icir.org/pushback/pushback-Jul01.pdf/.

[10] Network Simulator ns2, online documentation available at: http://www.isi.edu/nsnam/ns/.

[11] Altman, E.; Avrachenkov, K. and Barakat, C., “A Stochastic Model of TCP/IP with Stationary Random Losses ,” Proc. of SIGCOMM 2000, August 2000.

[12] Wu-chang, Feng; Shin K.G.; Kandlur, D.D. and Saha, D., “The BLUE active queue management algorithms,” *IEEE/ACM Transactions on Networking*, Vol. 10, No. 4, pp 513- 528, Aug 2002.

Appendix: proof of Lemma 1:

Denote $\Delta\lambda^x(t)$ to be the rate change of aggregate when it is perturbed with drop rate $x(t)$. By definition we will have:

$$E[\eta_{s_i^a}(r)] = \int_0^T s_i^a(t) E[\Delta\lambda^r(t)]\,\mathrm{dt}. \tag{18}$$

From the linearity assumption $E[\Delta\lambda^r(t)]$ in $r(t)$ we can conclude:

$$E[\Delta\lambda^r(t)] = E[\Delta\lambda^{r_i}(t)] + \sum_{j\neq i} E[\Delta\lambda^{r_j}(t)]. \tag{19}$$

Substituting (19) in (18) yields:

$$E[\eta_{s_i^a}(r)] = E[\eta_{s_i^a}(r_i)] + \sum_{j\neq i} E[\eta_{s_i^a}(r_j)]. \tag{20}$$

To complete the proof, it suffices to prove $E[\eta_{s_i^a}(r_j)] = 0$ for $j \neq i$. We have $r_j(t) = A_j s_j(t)$. Now we use the assumption that $r_j(t)$ changes slower than the aggregate response time. Hence $r_j(t)$ can be approximated by using a piecewise constant function. For an interval on which $r_j(t)$ is constant, the traffic aggregate responds and settles down to a value. In the next interval $r_j(t)$ jumps to a new value, and so $E[\Delta\lambda^{r_j}(t)]$ responds accordingly, and after experiencing a small transient time settles down to a new steady state value. According to the linearity assumption $E[\Delta\lambda^{r_j}(t)]$ on each interval is proportional to the constant value of $r_j(t)$ on that interval. This means that $E[\Delta\lambda^{r_j}(t)]$ tracks the piecewise constant shape of $r_j(t)$. So by ignoring the short transients of $E[\Delta\lambda^{r_j}(t)]$ at the beginning of each interval we will have:

$$E[\Delta\lambda^{r_j}(t)] \approx C_j r_j(t) = C_j A_j (s_j^d + s_j^a(t)) \tag{21}$$

in which s_j^d is the DC component of $s_j(t)$ over interval $[0, T]$. Recall

$$E[\eta_{s_i^a}(r_j)] = \int_0^T s_i^a(t) E[\Delta\lambda^{r_j}(t)]\,\mathrm{dt}. \tag{22}$$

Substituting (21) in (22) and using orthogonality assumption of (8) yields: $E[\eta_{s_i^a}(r_j)] = 0$. **QED**