Exogenous-Loss Aware Traffic Management in Overlay Networks

Toward Global Fairness

Mina Guirguis Azer Bestavros Ibrahim Matta

Computer Science Department, Boston University, Boston, MA 02215, USA

Abstract

For a given TCP flow, exogenous losses are those occurring on links other than the flow's bottleneck link. Exogenous losses are typically viewed as introducing undesirable "noise" into TCP's feedback control loop, leading to inefficient network utilization and potentially severe global unfairness. This has prompted much research on mechanisms for hiding such losses from end-points. In this paper, we show that low levels of exogenous losses are surprisingly beneficial in that they improve stability and convergence, without sacrificing efficiency. Based on this, we argue that exogenous-loss awareness should be taken into account in overlay traffic management techniques that aim to achieve global fairness. To that end, we propose an *eXogenous-loss aware* Queue Management (XQM) approach that actively accounts for and leverages exogenous losses on overlay paths. We envision the incorporation of XQM functionality in Overlay Traffic Managers (OTMs). We use an equation based approach to derive the quiescent loss rate for a connection based on the connection's profile and its global fair share. In contrast to other techniques, XQM ensures that a connection sees its quiescent loss rate, not only by *complementing* already existing exogenous losses, but also by actively *hiding* exogenous losses, if necessary, to achieve global fairness. We establish the advantages of exogenous-loss-aware OTMs using extensive simulations in which we contrast the performance of XQM to that of a host of traditional exogenous-loss unaware techniques.

Key words: Service Overlay Networks, Traffic Management, TCP Models, Control Theory, Transient Analysis, Simulations.
PACS: Performance Evaluation, Wireless Networks, Network Modeling and Analysis, Network Operation and Management, Congestion and Flow Control.

[⋆] This work was supported in part by NSF grants ANI-0095988, ANI-9986397, EIA-0202067 and ITR ANI-0205294, and by grants from Sprint Labs and Motorola Labs.

1 Introduction

One of the defining characteristics of the Internet is that it caters to an increasingly heterogeneous set of constituents. As such, a network resource is likely to be shared by flows with significantly different characteristics. While some may command fairly long RTTs as a result of traversing a satellite link, others may be subject to multiple congestions as they traverse a large number of hops with bursty cross-traffic, and worse yet, others may be subject to excessive losses as they traverse noisy wireless channels. To deal with this heterogeneity, networking research has traditionally focused on mitigating the sources of heterogeneity in a piecemeal approach. Examples of this are abound: from wireless TCP research that attempts to contain wireless losses [1,2], to research on new rate adaptation mechanisms that are suitable for high bandwidth-delay product networks [3–5]. While dealing with such issues separately leads to simpler "specialized" solutions to different problems (*e.g.*, wireless losses, large bandwidth-delay product flows), it is not clear if *such solutions may be working at cross purposes from one another.* In this paper we identify one such instance–namely, the impact of exogenous losses on TCP's performance and the advantages of leveraging such losses in an overlay setting to improve stability, efficiency, and fairness.

Motivation: Packet loss (or marking) events are interpreted by end-to-end transmission control mechanisms (such as TCP) as constituting the "feedback" signal from the bottleneck link to which such a mechanism must adapt its sending rate. As such, packet losses which are not incidental to that link pose a formidable challenge to an end-to-end transmission control protocol's ability to claim its fair share of network resources and/or react effectively to changes in network resource availability. In an overlay setting, the node at the entry point of an overlay link (henceforth we refer to such node as *Overlay Traffic Manager*, or OTM for short) would manage the capacity of its overlay link as well as its *local* buffer. It is very likely in this case that some losses occur somewhere else at any of the underlying physical (underlay) links that make up the overlay (logical) link to another OTM. In this paper, we use the term *exogenous losses* to refer to such losses. Exogenous losses can be thought of as occurring in a manner that is independent of the source's short-term behavior or its long-term fair share of network resources. The emergence of exogenous losses could be attributed to two radically different causes: the first is simply a consequence of traversing lossy channels (*e.g.*, wireless hops, satellite links), whereas the second is due to the bursty nature of cross-traffic on non-bottleneck links. The magnitude of the exogenous losses present and observed by a flow depends largely on the characteristics of the path traversed. One would expect the magnitude to have a wide variance due to the heterogeneity of the Internet.

Exogenous losses are problematic as they constitute "noise" with which a transmission controller must reckon. Unchecked, exogenous losses could be quite harmful. By preempting a source from claiming its fair share of the available bottleneck link capacity, exogenous losses may result in an unfair allocation of the bandwidth of overloaded links, or in a decreased utilization of underutilized links. Moreover, unwarranted reactions to exogenous losses may jeopardize stability and convergence properties. Recent research efforts have started to address these issues by adding specialized functionality either *in the middle* or/and *at the end-points* of the network. For example, through the use of a TCP proxy, losses on a wireless link could be hidden from end-points [1,2]. Alternatively, the negative impact of exogenous losses could be mitigated by enabling a source to diagnose the cause of packet losses and to react differently to different types of losses [6–9,3], or by slowing down its reaction to packet losses through the use of "smoother" control rules [10].

For the purposes of this paper, we focus our attention on the first of the above-mentioned negative implications of exogenous losses—namely their impact on *global fairness*. The bandwidth allocated to a flow is globally fair if it reflects the fair share of the capacity of the bottleneck link for that flow, in either an absolute or relative sense, e.g., w.r.t. Round Trip Time (RTT).

Overview and Contributions: While countering the effects of exogenous losses is a worthy goal, a more important goal is to assess the extent to which these losses actually impact the behavior of control loops. More to the point, to be able to assess the usefulness of the plethora of traffic control strategies dealing with effects of exogenous losses, we need a rigorous methodology for the analysis of the emergent behaviors that result from the composition of end-to-end protocols (*e.g.*, increase-decrease rules), network element behaviors (*e.g.*, RED/AQM (Active Queue Management) [11]), and new application-level functionalities. To that end, a particularly promising approach is to marshal techniques from control theory and optimization theory to the modeling and evaluation of complex network transmission control strategies, as exemplified in a number of recent efforts [12,4,13]. While useful, these efforts were limited by the fact that they did not explicitly model exogenous losses.

In this paper, we capture the effect of exogenous losses by extending a dynamic fluid model of the widely deployed Transmission Control Protocol (TCP) [14]. As one would expect, we show that high levels of exogenous losses lead to inefficient network utilization and potentially severe global unfairness. Surprisingly though, we also show that *low levels of exogenous losses introduce convergence to fairness properties that are both beneficial and desirable!* Specifically, we show that if exogenous loss levels do not adversely affect global fairness (*i.e.*, they do not exceed the value necessary for a flow to converge to its global fair share dictated by its bottleneck link), they tend to improve stability and convergence, without sacrificing efficiency. We elaborate on this point below.

Since TCP, by its nature, adaptively seeks available bandwidth, exogenous losses in effect impose an upper limit on achievable TCP throughput. The extent to which exogenous losses limit achievable throughput makes the crucial difference between desirable and undesirable exogenous losses. In particular, if this limit lies below a connection's long-term fair share, then exogenous losses cripple that TCP connection. Otherwise, we show that exogenous losses enable the fast and stable convergence of TCP connections to their long-term fair shares of network resources. This is because such exogenous losses serve as early error notifications to the sources, which, similar to RED (Random Early Detection) [11], randomize packet drops across all connections. This randomness prevents an individual TCP connection from monopolizing the bottleneck resource, in addition to preventing several connections from synchronizing their sending behavior which may result in high delay variance (jitter). Thus, low levels of exogenous losses, which do not force TCP throughput to dip below its long-term fair share, can be beneficial in reaching an *efficient*, *stable* and *fair* allocation of resources.

This observation suggests that the common wisdom of utterly hiding all exogenous losses may indeed be counter-productive. Even if such hiding is harmless, the overhead of implementing it—for example through local error recovery over wireless access links using Snoop [2]—may not be justified. This observation also suggests that it may be *beneficial* to "use" exogenous losses for traffic management purposes in overlay settings. Namely, to ensure global fairness, Overlay Traffic Managers (OTMs) [15–17] must be *exogenous-loss aware* in that they must hide exogenous losses only when they exceed a certain nominal value corresponding to said globally fair allocation of the link's available bandwidth.

Towards a constructive application of our findings, we argue that exogenous-loss awareness should be taken into account in overlay traffic management systems. In particular, we propose an *eXogenous-loss aware* Queue Management (XQM) approach that actively accounts for and leverages exogenous losses already introduced by other processes in the network underlay. Three of these schemes are demonstrated in [22]. In this paper, we focus on one instantiation that can be regarded as a per-flow (or per-class) implementation of REM [18] at OTMs, with the exception that its action is dictated by the quiescent loss rates necessary to achieve global fairness among flows in the presence of exogenous losses. Note that this per-flow or per-class state needs to be maintained only for flows/classes that are *active in the overlay network*—we expect their number not to be large.

Our goal in this paper is *not* to develop yet another AQM (albeit used at an overlay node), but rather *our goal is to introduce the concept of "exogenous-loss awareness" and demonstrate its benefits to overlay traffic management.* We envision the deployment of XQM-enabled OTMs at network boundaries—

maintaining a profile for each long-lived flow (or flow aggregate) passing through it. This profile includes estimates of the current connection's throughput. We use an equation based approach to derive the quiescent packet loss rate to impose based on the connection's profile and its fair share (allocated by the overlay OTM node). In contrast to other possible OTM queue management techniques, XQM ensures that a connection sees its quiescent loss rate, not only by *complementing* already existing exogenous losses, but also by actively *hiding* exogenous losses, if necessary, to achieve global fairness.

Note that under an exogenous-loss *unaware* fair-queueing/scheduling scheme (reviewed in Section 2), a TCP flow may also fail to reach its allocated rate in the presence of (additional) exogeneous losses. Although our XQM agent is inspired by AQM techniques, the extension of fair-queueing/scheduling techniques to also become exogenous-loss aware is also possible but we do not investigate such FQ extensions in this paper. *We note that it is not a matter of fine-tuning existing AQM or FQ algorithms—it is their unawareness of (additional) exogeneous losses that hinders their ability to maintain global fairness.*

To illustrate how an XQM agent deployed at an OTM could leverage exogenous losses, consider a TCP flow for which a quiescent 2% loss rate would result in a global fair share. If exogenous losses amount to 1%, then the XQM agent would introduce additional losses to bring the total loss rate to 2%. [1] On the other hand, if exogenous losses amount to 4% then the XQM agent could leverage any number of mechanisms to hide up to half of these losses to bring the total loss rate down to 2%.

We establish the advantages of exogenous-loss awareness using extensive simulations in which, we contrast the performance of an XQM agent to that of a host of traditional exogenous-loss unaware traffic management agents at OTMs.

Paper Outline: The rest of the paper is organized as follows. We motivate this work by presenting relevant related work in Section 2 and also throughout the paper, when appropriate. In Section 3, we present a dynamic model of TCP that incorporates exogenous losses. We analytically derive a lower bound on losses that need to be hidden from TCP sources to ensure efficient operation. In Section 4, we discuss the effects of exogenous losses on the behavior of TCP connections. We capitalize on this in Section 5, where we outline and evaluate the performance of our XQM overlay traffic management approach. Section 6 presents XQM's performance evaluation compared to other buffer manage-

[1] Had exogenous losses been hidden through an independent mechanism elsewhere (*e.g.*, using Snoop), new losses would have had to be introduced. This is a perfect instance of what we mentioned earlier regarding solutions working at cross purposes from one another.

ment approaches (inspired from the AQM literature) in different setups. We conclude in Section 7 with a summary of results.

2 Related Work

The work we present in this paper relates to a fairly large body of networking literature, targeting the goal of improving efficiency and fairness of transmission control loops. We exemplify the various flavors of this body of work below.

Control-Theoretic Modeling and Analysis: Marshaling techniques from control and optimization theory has been a fruitful direction as evidenced by a number of results, exemplified by the works in [30,13,26,31,19,32–35]. In that respect, we single out the works in [12,13], which investigated the stability regions for TCP over RED using a dynamic fluid model. Kelly *et al.* [19] model TCP/AQM as an optimization problem, where the maximization of the aggregate resource utility is sought. These techniques, however, did not explicitly model exogenous losses. Rather, they focused mostly on congestion control.

Active Queue Management/Scheduling Schemes: Overlay traffic management is conceptually similar to (albeit at a higher layer) AQM schemes (e.g., [11,26,35,36]). AQM designs have focused on the management of network congestion, with queue/buffer stabilization as the primary goal. Other schemes—notably FRED [27]—took a more active approach to protect flows that are particularly vulnerable (if additional losses are imposed) due to excessive shrinkage in buffer occupancy. As a byproduct of this special protection of vulnerable flows, FRED protects flows that are subject to excessive exogenous losses from further damage as they go through it. However, it is important to note that FRED's protection of such flows is triggered by inadequate throughput (as opposed to an explicit accounting and management of exogenous losses) to protect them from excessively poor performance (e.g., due to the incidence of timeouts). On the other hand, schemes like fair queuing [37] and GPS [38], with per-flow state can easily provide *local* fairness, while our scheme is working towards *global* fairness. Clearly, the presence of exogenous losses negatively impacts the performance of all these schemes, since they are unaware of (and not equipped to counteract the effects of) such losses.

Explicit Treatment of Exogenous Losses: Dealing with exogenous losses explicitly was addressed in projects that considered the impact of wireless communication (and wireless drops in particular) on TCP. A number of studies proposed breaking the transmission control loop into two segments [1], thus "hiding" the exogenous, wireless losses from the "wired" segment of the connection (not to mention "breaking" the end-to-end semantics of TCP). Other schemes (e.g. Snoop [2]) attempt to hide all wireless losses using local retransmission at the wireless access point. Another set of studies opted to "hide" exogenous losses by assigning the task of dealing with such losses to end-hosts, whereby the sender is empowered with diagnostic functionality that enables it to infer the reason for a packet loss and to react accordingly. Examples of this line of work are given in [6–9]. More recently, and in order to avoid the severe implications of "overacting" to non-congestion-induced (e.g., exogenous) losses in high-speed, long-latency networks, the work in [3] suggests transmission control rules that use additional predictors (e.g., queuing delays) to moderate the reaction of senders to such losses. For both of these approaches (dealing with exogenous losses in the middle or at end-points), exogenous losses are regarded as noise that must be completely eradicated (or hidden) from senders. None of these techniques advocate that some level of exogenous losses is harmless–let alone beneficial to boosting fairness and stability. And, clearly, none of these techniques leverages exogenous losses in the communication of the feedback signal from the bottleneck link.

3 Modeling TCP + Exogenous Losses

In this section, we extend an analytical fluid model, similar to that proposed in [13,19,12,20], to capture the effect of exogenous losses on closed-loop TCP control loops. We present ns-2 [21] simulations to validate our observations from the model.

3.1 Model Derivation

We consider a dynamic fluid model of m TCP connections traversing a single bottleneck overlay node whose link capacity is C. The round trip time $r_i(t)$ at time t for connection i is equal to the round-trip propagation delay D_i between the sender and the receiver for connection i, plus the queuing delay at the bottleneck node. Thus $r_i(t)$ can be expressed by

$$r_i(t) = D_i + \frac{b(t)}{C} \tag{1}$$

where $b(t)$ is the backlog buffer size at time t at the bottleneck node. We denote the propagation delay from sender i to the bottleneck by D_{s_ib}, which is a fraction α_i of the total propagation delay.

$$D_{s_i b} = \alpha_i D_i \tag{2}$$

The backlog buffer $b(t)$ evolves according to the equation:

$$\dot{b}(t) = \sum_{i=1}^{m} x_i(t - D_{s_i b}) - C \tag{3}$$

which is equal to the input rate $x_i(.)$ from the m connections minus the output link rate. Notice that the input rates are delayed by the propagation delay from the senders to the bottleneck $D_{s_i b}$.

We assume that the links between the bottleneck and the receivers are subjected to exogenous packet losses, and that all connections see the same level of exogenous losses. It follows that the total packet loss probability $q(t)$ observed by senders would comprise the congestion-induced loss probability $p_c(t)$ (due to buffer overflow at the bottleneck) as well as the exogenous loss probability $p_e(t)$. Thus, the total loss probability seen by senders is given by

$$q(t) = 1 - (1 - p_c(t))(1 - p_e(t)) \approx min(p_c(t) + p_e(t), 1) \tag{4}$$

where the congestion loss probability $p_c(t)$ depends on our choice of a queue management implementation at the bottleneck node.

For DropTail, $p_c(t)$ is simply given by

$$p_c(t) = \begin{cases} 0 & b(t) < B \\ 1 & b(t) = B \end{cases} \tag{5}$$

where B is the maximum buffer size. [2]

An overlay node needs to manage its capacity and buffer much in the same way as an underlay router would. If we assume that the bottleneck OTM node manages its local buffer using RED [11], the congestion loss probability $p_c(t)$ is given by [3]

[2] We assume that when operating in a certain regime at time t, *e.g.*, when $b(t) < B$, the probability that the queue is full is small enough that the queue length is practically less than B over all sample paths. This assumption is validated by the ns-2 simulations presented later in this section.

[3] For simplicity, we follow the same assumptions of other studies by ignoring the uniformization of packet drops [11]. This assumption is relaxed in our ns-2 simulations.

$$p_c(t) = \begin{cases} 0 & v(t) \leq B_{min} \\ \sigma(v(t) - \varsigma) & B_{min} < v(t) < B_{max} \\ 1 & v(t) \geq B_{max} \end{cases} \tag{6}$$

where σ and ς are the RED parameters given by $\frac{P_{max}}{B_{max}-B_{min}}$ and B_{min}, respectively, and $v(t)$ is the average queue size, which evolves according to the equation:

$$\dot{v}(t) = -\beta C(v(t) - b(t)), \qquad 0 < \beta < 1 \tag{7}$$

Notice that in the above relationship, we multiply β by C since RED updates the average queue length at every packet arrival, whereas our model is a fluid model [13,12].

The throughput of TCP, $x_i(t)$ is given by

$$x_i(t) = \frac{w_i(t)}{r_i(t)} \tag{8}$$

where $w_i(t)$ is the size of the TCP congestion window for sender i.

According to the TCP Additive-Increase Multiplicative-Decrease (AIMD) rule, the dynamics of TCP throughput for each of the m connections can be described by the following differential equations:

$$\dot{x}_i(t) = \frac{x_i(t - r_i(t))}{r_i^2(t)x_i(t)}(1 - q(t - D_{bs_i}(t))) - \frac{x_i(t)x_i(t - r_i(t))}{2}(q(t - D_{bs_i}(t))) \\ i = 1, 2, .., m \tag{9}$$

The first term represents the additive increase rule, whereas the second term represents the multiplicative decrease rule. Both sides are multiplied by the rate of the acknowledgments coming back due to the last window of packets $x_i(t - r_i(t))$. Thus, for positive acknowledgments that are arriving at a rate of $x_i(t - r_i(t))(1 - q(t - D_{bs_i}(t)))$, the window $w_i(t)$ is increased by $\frac{1}{r_i(t)}$ and for the negative acknowledgments that are arriving at a rate of $x_i(t - r_i(t))q(t - D_{bs_i}(t))$, the window, $w_i(t)$ is halved. In the above equations, the time delay from the bottleneck to sender i, passing through the receiver i, is given by

$$D_{bs_i}(t) = r_i(t) - D_{s_ib} \tag{10}$$

The above analytical model captures the essential dynamics necessary to gain

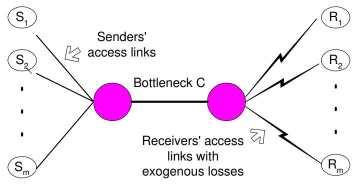


Fig. 1. Dumbbell topology used in numerical evaluation and simulations.

valuable insights. We later validate the model using more detailed simulation experiments.

3.2 Model Application

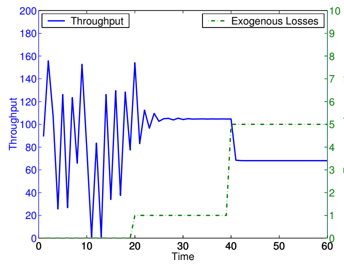
Low *et al.* [12] studied the dynamics of TCP over RED queues through linearization around equilibrium points. [4] While useful, linearization fails to track the system trajectories across different regions dictated by the non-linear equations.

We refer the reader to [22], where we show the linearization of the system (TCP + exogenous losses) modeled above. In particular, we show how such a system switches between an open-loop control, when exogenous losses are high, and a closed-loop control, otherwise. This switching between operating regions prevents us from using traditional transient control analysis. Thus, in the remainder of this section, we solve the above set of non-linear equations numerically for a careful and continuous tracking of the model's behavior through different operating regions.

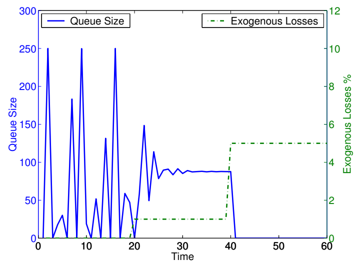
3.3 Impact on Efficiency

Figure 1 depicts the topology under consideration. We set the total number of competing connections to 20; we set the capacity C to 2,000 $\frac{pkts}{sec}$; and we chose the propagation delay of all connections uniformly at random between 80 and 120 msec. Each connection's fair share of the link is around 100 $\frac{pkts}{sec}$. The total buffer size at the bottleneck is chosen to be 250 packets. RED's minimum and maximum buffer thresholds are set to 50 and 120 packets, respectively. The weight parameter β was set to 0.00001 and P_{max} was set to 0.1. We also chose α_i in equation (2) uniformly at random in the interval [0.25-0.5]. During the time period [0, 20) we introduce 0% exogenous losses, during [20, 40) the rate of exogenous losses is increased to 1% and finally during [40, 60], exogenous losses are increased further to 5%.

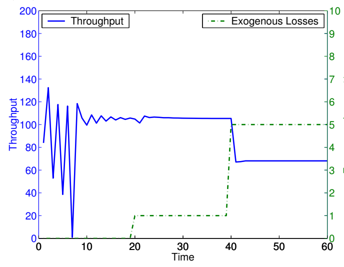
[4] Linearization assumes (and hence requires) that the system always stays within a certain operating regime.



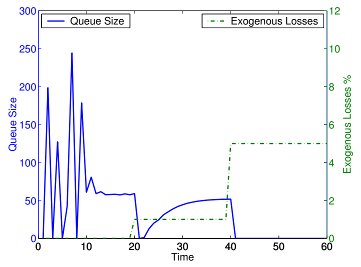
(a) DropTail: Throughput



(b) DropTail: Queue Size



(c) RED: Throughput



(d) RED: Queue Size

Fig. 2. Impact of exogenous losses (0%, 1%, and 5%) on efficiency of DropTail (top) and RED (bottom) as reflected by throughput (left) and buffering (right).

Figure 2 shows the throughput and the queue size obtained using our numerical solution under both DropTail (top row) and RED (bottom row). [5]

In the first 20 seconds, *i.e.*, under zero exogenous losses, TCP throughput oscillates between low and high sending rates for DropTail, while RED sustains these oscillations only until the average queue size reaches its steady-state value (around time 10). In the next 20 seconds, when the level of exogenous losses increases to 1%, TCP throughput converges (perfectly) to its fair share under both DropTail and RED. Notice how the queue size converges to a steady-state (non-zero) value, hence the system is well utilized. In the last 20 seconds, exogenous losses (now increased to 5%) result in the convergence of each $x_i(t)$, albeit to a value lower than the fair share and the queue size drops to zero, hence the system is under utilized. This observation suggests that low levels of exogenous losses (e.g., 1%) do not degrade the throughput of TCP. But clearly, when exogenous loss rates are increased significantly (e.g., 5%), TCP's throughput suffers and the system becomes under utilized (e.g., below the fair share of 100 $\frac{pkts}{sec}$).

A transmission control loop is said to be *efficient* if the TCP throughput for that loop matches the bottleneck link capacity. Thus, at steady state, the following two equations should be satisfied for an efficient network utilization. These equations are obtained by setting the derivatives to zero in equations (3) and (9).

[5] We later consider other AQM techniques and investigate their vulnerabilities to exogenous losses.

$$\sum_{i=1}^{m} \hat{x}_i = C \tag{11}$$

$$\hat{x_i} = \frac{1}{\hat{r_i}}\sqrt{2(\frac{1}{\hat{q_i}} - 1)} \tag{12}$$

Clearly, the steady-state TCP throughput $\hat{x_i}$ is inversely proportional to the square root of the total loss probability $\hat{q_i}$, which in turn is directly affected by the exogenous loss rate $\hat{p}_e$.[6] For a steady-state behavior, $\hat{q_i}$ must be larger than zero. Having no drops removes the upper limit on the rate/window and this, in theory, will cause it to grow indefinitely.

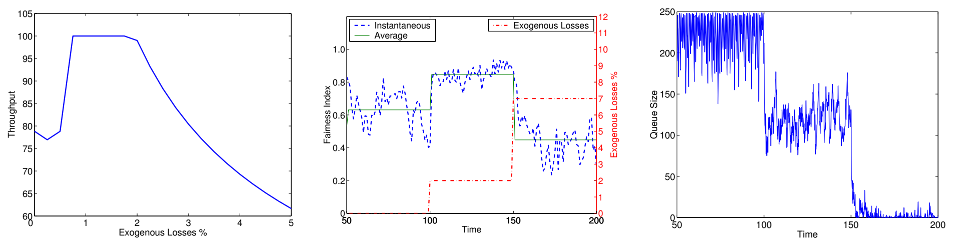
As the steady-state value of $\hat{p}_e$ increases, the sending rate would start to decrease, approaching zero. This could prevent TCP throughput $\sum_{i=1}^{m} \hat{x}_i$ from reaching C, i.e. equation (11) cannot be satisfied. The value of $\sum_{i=1}^{m} \hat{x}_i$ being less than C means that the system is under utilized. Hence TCP is forced to operate with no buffering at the bottleneck, and no congestion signals going back to senders. When this happens, the TCP transmission control loop is actually broken—it operates as an *open-loop control system* with no feedback from nodes.

When exogenous losses are not present, nothing hinders the increase of TCP throughput so as to match its bandwidth share.[7] Once the connection hits its bandwidth share, packets start to accumulate until $b(t)$ reaches B under DropTail, or the average queue size starts building up until it exceeds B_{min} under RED. At that time, congestion signals are generated and the sender would back off and this cycle repeats (cf. equation (9)).

The presence of exogenous losses imposes an upper limit on TCP's throughput and it is crucial *where* this upper limit lies. If this upper limit is close to the connection's long-term fair share, then these exogenous losses turn out to improve the connection's convergence to its fair share. This is exactly what happens in the time period [20, 40) in Figure 2. Without such exogenous losses in [0, 20), the connection's throughput shows large oscillations under DropTail.

[6] Observe that equation (12) resembles the so-called TCP-friendly equation [23], except that in our model, $\hat{q_i}$ is not necessarily a Bernoulli probability, but depends on queue management parameters.

[7] The connection is still limited by its round-trip time, but eventually will hit its bandwidth share. We assume that connections are not limited by the advertised receiver's window.



(a) (b) (c)

Fig. 3. Model-predicted effect of exogenous losses on efficiency (left) and validation via simulations of effect on fairness (middle) and queue size (right) under DropTail.

Consider the same setup described above with DropTail, except that all connections have an identical propagation delay of 100 msec. If the goal is to allocate an equal share of the bandwidth to each TCP connection, then using equation (11), each connection's (long-term) fair share, $\hat{x}_i$, would equal 100 $\frac{pkts}{sec}$. Equation (12) can be solved for $\hat{q}_i$ for a given round-trip time $\hat{r}_i$, which depends on the steady-state buffer occupancy. Indeed, Figure 3(a) shows that when exogenous losses are in the range of 1% to 2%, the throughput converges to the fair share value. Below or above these values, the fair share is not matched due to either oscillations (left) or under utilization (right), respectively.[8] Notice that exogenous losses around 2% represent the case where the steady-state round-trip time is equal to the propagation delay with zero buffer occupancy.

To validate the above observations, we conducted a simple ns-2 simulation on a simple dumbbell topology similar to the one in Figure 1. The bottleneck link capacity is set to 16Mb and its propagation delay is set to 1 msec. A total of 20 TCP connections are created between the senders and the receivers. Senders and receivers connect to the nodes through access links with propagation delay chosen uniformly at random between 1 and 4 msec. The receivers' access links are associated with error modules that would represent the effect of exogenous losses. At time 0, we start with no exogenous losses, at time 100 we set the exogenous losses to 2% across all receivers' access links, at time 150, exogenous losses are increased to 7% and the experiment ends at time 200. We use DropTail at the bottleneck link and we ignore the first 50 seconds of the simulation experiment. Figures 3(b) and 3(c) show the effect of exogenous losses on fairness and on queue size, respectively. Fairness is computed across all connections every 1 second interval, using Chiu and Jain's Fairness Index [24], which is given by:

[8] The oscillations in the fluid model are a by-product of synchronization effects among flows since *all* of them react to the same error signal. This effect is less pronounced in simulations.

$$f(x_1, x_2, x_3, .., x_m) = \frac{(\sum_{i=1}^{m} x_i)^2}{m \times \sum_{i=1}^{m} x_i^2} \tag{13}$$

Notice how the fairness index improved significantly when exogenous losses increased to 2% since such value in effect helped the connections converge to their fair share, and also helped the buffer size converge. Increasing exogenous losses to 7% leads to a deterioration in the fairness index and leads to under utilization of the network since the queue size gets closer to 0.

Many protocols have been developed for hiding *all* exogenous losses from the sender [2,1]. For example, in Snoop [2] the connection between the server and the client is in effect intercepted by an OTM proxy. This proxy buffers data packets to allow link-layer retransmission when duplicate acknowledgments, indicating packets lost over the wireless proxy-client link, arrive at the proxy. Snoop does not allow such duplicate acknowledgments to pass back to the sender to prevent it from doing fast retransmit and recovery (*i.e.*, halving its sending rate). While such OTM protocols attempt to improve efficiency by removing the upper limit imposed on throughput by exogenous losses, they could be hindering the convergence to fairness! Furthermore, hiding further packet losses from connections that are already getting their fair share would not be beneficial, but would only add the overhead of complete hiding (*e.g.*, the cost of buffering and local retransmission at the Snoop proxy)–not to mention the fact that another process (in this case an AQM) would have to reintroduce packet losses.

Ideally, we would like to always report a value of $\hat{q}_i$ to sender i that corresponds to its fair share, since this would mean that the network is utilized efficiently while, at the same time, connections have a fair chance to compete. In the next section, we address the challenges behind active management of exogenous losses in an overlay setting to achieve this goal.

4 Active Tuning of Exogenous Losses

As discussed in the previous section, for given bottleneck link and RTT characteristics, there exists a desirable value for the loss rate that would promote both convergence and efficiency of a TCP control loop. We use the term *quiescent loss rate* to refer to this desirable value. For example, a quiescent loss rate of 2% yielded both efficiency and convergence for the experimental setting used in Figure 3(a). In this section we examine the advantages and disadvantages of alternative active (AQM-inspired) approaches for relaying such quiescent loss rates from an overlay traffic manager to senders.

Exogenous Loss Unaware Signaling (Local Fairness): An overlay traffic manager (such as traffic shapers) may simply ignore exogenous losses by imposing a loss rate value that improves some local metric (*e.g.*, stability of the buffer backlog). An example of such a technique is RED (or other variants thereof, *e.g.*, [25]), whereby the dropping or marking of packets is conditioned on the local queue occupancy in order to stabilize the queue. As evident from the results in Figure 2(d), RED exhibits transient inefficiencies when faced with variability in exogenous loss rates over short time scales. Specifically, at time 20, TCP's queue size drops to zero, before converging again to the new steady-state value of around 50. This transient anomaly is less pronounced under DropTail.

The undesirable transient behavior exhibited under RED is due to RED's unawareness of exogenous losses, which is exacerbated by the lag time necessary for the average queue size (seen by RED) to reflect the "real" conditions. To elaborate on this, consider the case when RED's average queue length is above B_{min}. Now consider a situation whereby TCP flows react to a sudden increase in the exogenous loss rate (by backing off), which in turn would cause the RED queue to drain. Since RED uses the *average* queue length as an indicator of congestion, it would take RED some time to realize this new "drained" state. As a result, RED would keep on generating congestion signals (by dropping or marking packets) according to the stale higher value of its average queue length—causing further degradation in efficiency. [9] Obviously, under such conditions, the congestion-minded design of RED is challenged by exogenous losses, since it is no longer true that the sender reduces its rate only in response to congestion signals! As soon as the average queue size catches up with the new value below B_{min}, RED ceases to send its feedback signal. At that point, TCP is in fact operating as an open-loop control system and starts to increase its sending rate.

The inability of RED (as a representative of exogenous-loss unaware traffic management) to cope with exogenous losses is further complicated by issues of heterogeneity in flow characteristics (*e.g.*, the possibly wide range of exogenous loss rates across flows or aggregates thereof). Clearly, no traffic management would be able to address issues of global fairness without some accounting of flow characteristics. Indeed, if RED were to achieve global fairness, it would require more than parameter tuning, namely awareness of the presence of these losses and invoking the right control rules.

Exogenous Loss Aware Signaling (Global Fairness): The above discussion suggests that, towards global fairness, it is crucial for a traffic management approach to take into account the presence of exogenous losses. One possibil-

[9] This phenomenon was noted in [26], prompting the need for decoupling the queue size from the dropping/marking probability. We later examine the degree of vulnerabilities of more recent AQM techniques to exogenous losses.

ity is to have overlay traffic managers use a technique similar to FRED [27]. FRED accounts (albeit implicitly) for exogenous losses by protecting fragile flows—flows with small window sizes. It is important to note that accounting for exogenous losses is more of a side effect than a "by design" feature since FRED protects flows with excessively small window sizes, independent of whether flow fragility is due to exogenous losses, or simply a reflection of that flow's fair share. In contrast, our XQM approach presented later in this paper, makes the decision of when to introduce losses, when to hide losses, and when not to interfere, based on the level of exogenous losses present. In effect, an XQM agent in an OTM node utilizes such external losses, toward its own feedback signal. Thus, it provides the minimum interference and only when needed.

Assuming that the exogenous loss rate for a flow can be relayed to (measured/estimated by) an XQM-enabled overlay traffic manager, then it is possible to ensure that the sender will only see the quiescent end-to-end loss rate by having the XQM agent adjust its own control rules accordingly. Namely, if exogenous losses are below the quiescent rate, then it is possible to "introduce" losses to promote efficient convergence to a fair share. This could be done through randomly dropping or marking packets. If exogenous losses are higher than the quiescent rate then it would be necessary to "hide" such losses from the sender. This could be done in many ways, including link layer retransmission, forward error correction techniques, or replication over multiple paths (i.e. dispersity routing [28]) across the overlay network.

In practice, requiring that an XQM be able to obtain an accurate estimate of exogenous loss rates whether explicitly from or through measurement of network underlays is hard to achieve.[10] Thus, alternately, an XQM could dynamically adjust its behavior (*i.e.*, figure out the exact levels of losses to be introduced or hidden) in response to each connection's performance, without having to explicitly know the exogenous loss rate for such connections.

In the above discussion, we have assumed that exogenous loss rates are static. In a real setting, this is likely not to be the case. Thus, it is important to assess the impact of such variability, both over long and short time scales. We propose two different methods to massage the exogenous losses observable by senders, which we refer to as long-term adjustment and short-term compensation. Due to space limitation, we refer the reader to [22] for an evaluation of these techniques. Having demonstrated the benefits from exogenous aware signaling, we next turn our attention to how to design queue management schemes that are exogenous-loss aware.

[10] Recent "measurement from the middle" studies suggest that such approach may indeed be feasible [29].

5 eXogenous-loss aware Queue Management

In this section, we discuss and evaluate our proposed eXogenous-loss aware Queue Management (XQM) approach to traffic management in an overlay setting. An XQM agent's main goal is to tune the exogenous losses that are already present in the network to improve fairness of flows going through that agent, without sacrificing efficiency. Thus it provides each flow [11] with its fair share of resources. An XQM agent maintains a profile for each active flow it manages. This profile includes the current flow's throughput $x_i(t, MP)$, measured as the number of packets sent over the past *Measurement Period* (MP) and the current imposed loss rate $q_i(t)$. *Note that this per-flow (or per-class) state needs to be maintained only for flows/classes that are active in the overlay network—we expect their number not to be large.*

On a packet arrival, the XQM agent in the OTM node identifies the flow this packet belongs to and drops the packet with probability $p_i(t)$ that is a function of the current quiescent loss rate:

$$p_i(t) = \frac{q_i(t)}{1 - k \times q_i(t)} \tag{14}$$

where k is the number of packets queued since the last packet drop/mark. This insures that XQM spreads losses uniformly over time [11]. Every *Control Period* (CP), the XQM agent updates the current imposed loss rate $q_i(t)$ for each active flow. In [22] we propose three different schemes for implementing such an update (XQM ON-OFF, XQM PI-T and XQM PI-TB). We only present XQM PI-TB here; thus for the remainder of this paper, we use XQM and XQM PI-TB interchangeably.

XQM PI-TB: PI Throughput and Buffer Matching
In this implementation, two error signals are obtained by comparing the current throughput of the flow with its target fair share (allocated by the XQM agent) and the current buffer size with the target buffer size. Maintaining the buffer size at a low target ensures less jitter and shorter round-trip time. XQM PI-TB is thus given by: [12]

[11] For the purposes of this discussion, we don't insist on a strict definition of a flow. One can think of a 4-tuple definition (source IP, destination IP, source port #, destination port #), a 2-tuple definition (source IP, destination IP) or simply aggregates of these.
[12] This version of XQM can be regarded as a per-flow version of REM [18] where fair rates are explicitly allocated to flows so as to overcome the negative effects of exogenous losses.

$$q_i(t + CP) = q_i(t) + \delta(x_i(t, MP) - \hat{x}_i) + \phi(b(t) - \hat{b}) \tag{15}$$

where $b(t)$ is the current buffer size and $\hat{b}$ is the target buffer size. Unless otherwise specified, the allocated (fair) share of flow i, $\hat{x}_i$, is set by the XQM agent as simply an equal rate of the bottleneck capacity. As noted later, any weighted allocation of rates could also be applied. We also note that $\hat{x}_i$ determines the quiescent loss rate *dynamically* through equation (15), and hence there is *no* need for explicitly computing the quiescent loss rate from equation (12) (as in XQM ON-OFF [22]) which requires the estimation of RTT from the middle as in [29]. *The explicit estimation of exogenous loss rates is also not required.* Finally the four parameters CP, MP, δ and ϕ play a very important role in the general behavior of the XQM agent. We summarize our experience with these parameters next; details can be found in [22]. The default values used in our simulations were found to be quite robust over a wide range of scenarios.

First we focus on the interplay between the measurement and control periods. Because the control period (CP) directly affects the frequency with which the controller is invoked, we choose a small value (around 10 msec). Having a larger CP will cause the XQM agent to be less responsive, while having a smaller CP would only add to the overhead of invoking the controller. For a correct estimate of the connections' throughput, the measurement period (MP) should be a multiple of the congestion epochs. That is, it should be long enough to capture multiple packet drops. Having a shorter MP, will cause errors in throughput estimation due to window fluctuation. However, having a longer MP, will limit the XQM agent's ability to capture short-term behaviors. [13]

We now turn our attention to δ and ϕ, the weights in equation (15). These weights play an important role in the decision process. They specify the tradeoff between efficiency and fairness. In particular, a higher value of ϕ will tend to improve efficiency, while a higher value of δ will tend to improve fairness. There are four possible cases, which we consider next.

The first two of these cases are straightforward; they correspond to situations in which the two constituent controllers in equation (15) are in agreement as to whether the loss rate imposed on a flow is to go up or down. Namely, these two cases occur when the bandwidth (for a flow) and the buffer size are *both* below their prescribed values, or *both* above their prescribed values. Clearly, for the former, the XQM agent will decrease the loss rate imposed on the flow, and for the latter, the XQM agent will increase the loss rate imposed on the flow.

The other two cases correspond to situations in which the two constituent controllers in equation (15) are at odds with one another regarding whether

[13] Note that unlike REM, we are decoupling MP and CP.

the loss rate imposed on a flow is to go up or down. For example, what if a connection's throughput is less than the targeted throughput, but the buffer size is larger than the targeted buffer size? In our experiments (some of which we will present in the next section), we found that having a value of ϕ relatively larger than δ is helpful in cases when some connections are unable to get their fair share of the throughput (e.g., they are source limited). Under such conditions, we allow other connections to grab the available bandwidth by giving a higher "weight" to buffer-size matching. On the other hand, in our experiments, we found that having a value of ϕ that is much greater than δ tends to hurt connections that are already below their fair share (in an attempt to whip the buffer into matching its prescribed value). By tuning the values of δ and ϕ, the XQM agent is able to expose the tradeoffs between efficiency and fairness.

To summarize, an XQM agent at an OTM node has two key design features. The first is that the XQM agent decouples the measurement period from the control period. This decoupling allows the XQM agent to improve fairness (over longer time scales) without sacrificing efficiency (over shorter time scales). This is achieved by exercising control over short time scales based on throughput measured over longer time scales. The second key feature of the XQM agent is that it exposes the tradeoffs between efficiency (over shorter time scales) and fairness (over longer time scales). The selection of the characteristic time scales for measurement and for control, as well as the adjustment of the tradeoff between efficiency and fairness are *both* possible to manage dynamically based on the traffic profile. This dynamic tuning of the XQM agent's operation (based on traffic profiling) is the subject of a future paper.

6 Simulation Results

In this section, we present results from extensive ns-2 [21] simulation experiments we conducted to assess the advantages of exogenous-loss awareness. We do so by comparing an OTM that uses XQM with that using other exogenous-loss unaware approaches similar to those proposed for router-level AQMs—namely RED [11], FRED [27], REM [18] and PI [26].

6.1 Effect of Losses Due to Cross-traffic on Non-Bottleneck Links

Ideally, for a TCP connection to reach its fair share, it should get its loss signal from the bottleneck link only. In practice, due to network dynamics, a TCP connection could experience packet losses on (multiple) non-bottleneck links. The more congested/bursty path segments a connection traverses, the noisier the feedback signal (due to exogenous losses on non-bottleneck links). In this section, we show how an XQM agent in effect "adopts" packet losses on other underlay network links as its own—in effect using them towards the total packet losses it needs to impose on the flow. An XQM agent would decrease the losses it imposes as exogenous loss rates increase toward the quiescent loss rate, moreover it would increase the value it hides as exogenous loss rates increase above the quiescent loss rate. However, if the exogenous losses are close to the quiescent loss rate, the XQM agent will not interfere.

Figure 4 depicts the topology under consideration. We have three links AB, BC and CD of capacity 100 Mbps, 150 Mbps and 150 Mbps, respectively. For simplicity, all links have a one-way propagation delay of 1 msec. A total of 10 FTP connections, with unlimited data to send, traverse the *overlay* link from A to D. We refer to these as the overlay AD flows, with IDs from 1 to 10. In addition, three groups of 10 FTP connections, each representing cross-traffic with unlimited data to send, traverse exactly one of the links in the topology (*i.e.*, those underlay links making up the overlay link AD). We refer to these as the AB, BC and CD flows. Sources as well as receivers of these FTP flows connect to the overlay traffic managers at A and D, and to the routers at B and C, through separate "access" links.

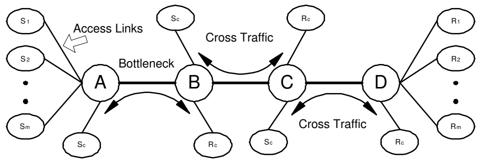
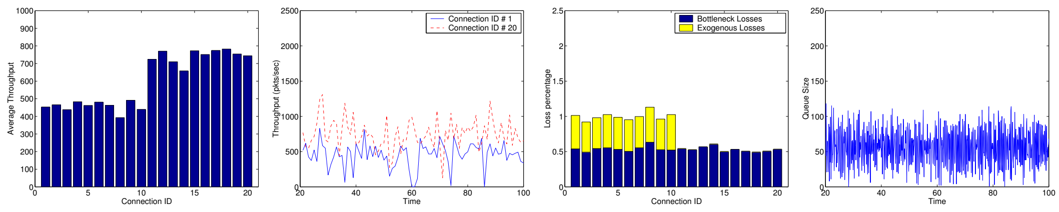


Fig. 4. Topology used in ns-2 simulation experiments. A-D represents an overlay link whereby an XQM agent at the OTM node A manages the capacity of the bottleneck.

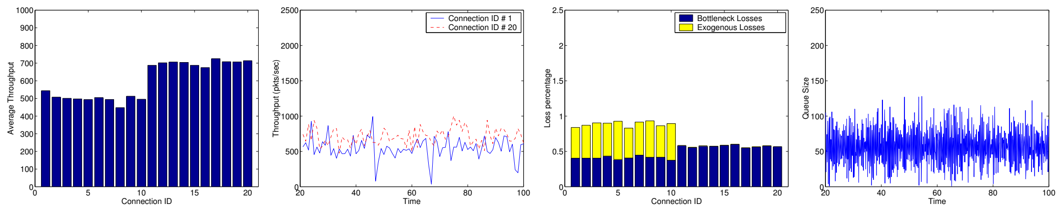
In the topology of Figure 4, the first link (AB) is the bottleneck of the 10 overlay AD flows and it uses the traffic management approach being evaluated–namely XQM, RED, FREQ, REM, or PI–whereas the second and the third (underlay) links (BC and CD) are managed using RED. [14]

Unless otherwise stated, in our experiments, we set RED's minimum and maximum buffer thresholds to 50 and 120 packets, respectively. The weight parameter β is set to 0.0001 and P_{max} is set to 0.1. The buffer size is chosen to be 250 packets at each link. All packets are 1,000 bytes in size. Also, unless otherwise stated, in our experiments, we set the parameters of the XQM agent at the OTM node A, CP, MP, δ and ϕ to be 0.01 msec, 10 seconds, 0.00001 and 0.00004, respectively.

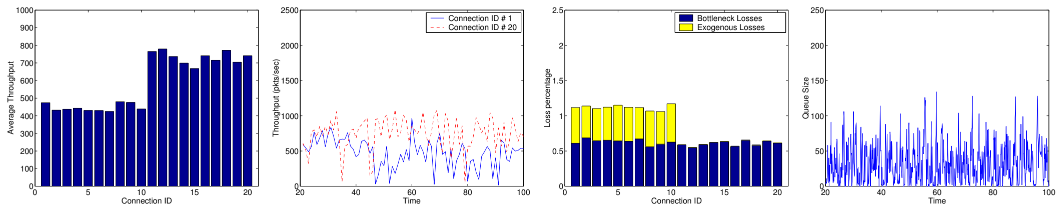
[14] Note that the bottleneck capacity of the overlay link does not have to be incident to the overlay node, A in this case. In general, the overlay node assumes its capacity to be the minimum capacity among all its underlay links. Several techniques exist to measure such bottleneck capacity over the overlay link.



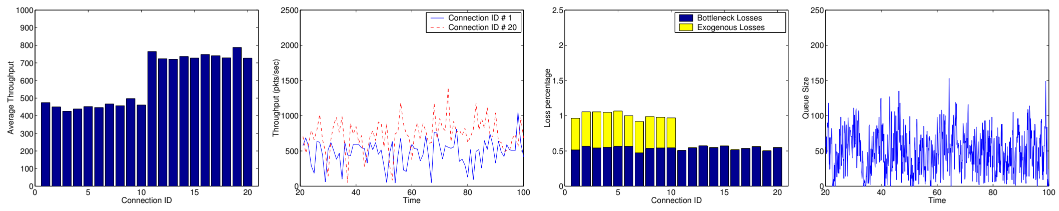
(a) RED



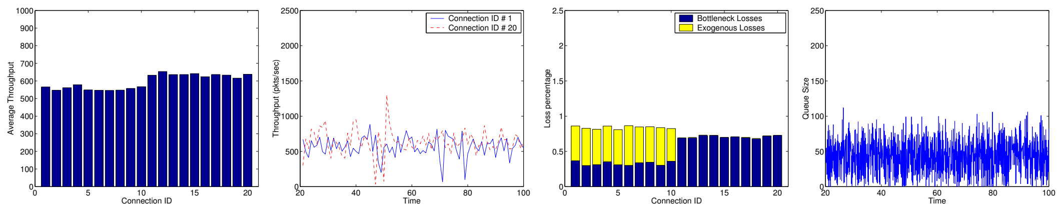
(b) FRED



(c) REM



(d) PI



(e) XQM PI-TB (CP $= 0.01$, MP $= 10$, $\delta = 0.00001$, and $\phi = 0.00004$)

Fig. 5. Comparative performance of various AQM approaches, showing long-term throughput for all flows (left), instantaneous throughput for two exemplary flows, average loss rates seen by all flows, and instantaneous buffer size (rightmost).

Experiment 1: We start with a simple case where all connections have the same round-trip time. To do so, we adjust the propagation delay on the access links so that all connections have the same round-trip time, taking into account the queuing delay at the (underlay) links BC and CD for overlay connections AD. We present the results across the first link (at the bottleneck overlay node A).

Figure 5 compares the performance of the different schemes–each shown in plots on separate rows. Plots in the first (leftmost) column represent the average throughput achieved by each connection, which is computed over the interval [20-100]. Plots in the second column represent the instantaneous throughput computed every one second interval. We only present two connections, one that belongs to the overlay set AD and another one that belongs to the set AB. Plots in the third column represent the average losses seen by each flow over the interval [20-100]. Overlay connections AD see losses that are also on other underlay links, i.e., "exogenous losses". Finally, plots in the last (rightmost) column represent the instantaneous queue size. In all plots, we ignore the first 20 seconds (for simulation warm-up purposes) and we calculate all metrics starting from time 20.

Despite the fact that all connections have the same round-trip time, overlay connections AD that traverse multiple congested links (i.e. connections 1 to 10) end up with less throughput. Since RED, REM and PI traffic management approaches apply the same loss rate across all connections, they don't compensate for any exogenous losses on other links. FRED, on the other hand, compensates a little bit as evident by the values of the loss rate for FRED. FRED applies lower loss rate values to connections AD and higher values to AB. Still, AD (overlay) connections can not reach their fair share.

On the other hand, the XQM PI-TB agent at the OTM node A applies just enough losses so that the total loss rate seen by any connection is the same—hence the better fairness delivered by XQM. Also, it maintains the buffer size at the target level of 50 packets.

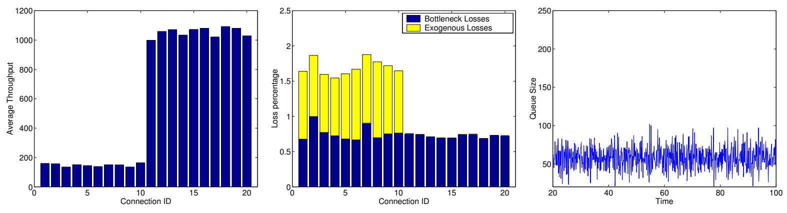
Experiment 2: In the previous experiment, we fixed the propagation delay so that all connections experience the same round-trip time, and thus giving us an opportunity to observe/study the effect of exogenous losses on multiple (non-bottleneck) hops on overlay connections AD. Now, we repeat the same experiment, except that all the access links (for connections AD as well as for cross traffic connections AB, BC and CD) have a propagation delay that is uniformly distributed between 5 and 10 msec. Now, connections AD have a longer round-trip time compared to AB, since they traverse more links. So two questions arise: how much worse would overlay connections AD fare? and how effective is the XQM agent in dealing with such scenarios?

Increasing the round-trip time for overlay connections AD would only make the situation worse (i.e., if they can not get their fair share with a shorter round-trip time, they will certainly not get their fair share with a longer one, due to the additional bias of TCP against connections with longer round-trip times). Indeed, this is the case under RED, FRED, PI and REM; overlay connections AD (with longer RTT *and* subjected to additional exogenous losses on underlay links) can not get their fair share. Due to space limitation, we only present the performance under RED (as a baseline) and FRED (which had the best performance among all other exogenous-loss unaware AQMs).

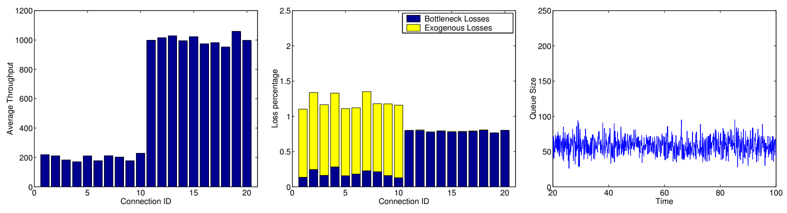
In our experiments we found that OTMs using REM and PI agents behave quite similarly to those using RED, imposing the same loss rate across all connections. FRED imposes the minimum losses, however it still taxes AD overlay connections, putting them at a disadvantage with respect to reaching their global fair share of bandwidth. The XQM agent, on the other hand, does not drop any packets from AD overlay connections (connections AD are thus only limited by exogenous losses on other underlay links). The plots in rows (a), (b) and (c) of Figure 6 show the performance of RED, FRED, and XQM PI-TB, respectively (also, the Fairness Index is noted). Since overlay connections AD are still limited by exogenous losses, despite the XQM agent ceased any drops, they fail to grab their fair allocation. That is why the buffer size in row (c) is below the target value of 50. Notice that as far as the XQM agent is concerned (for now), it is not imposing any losses on overlay connections AD. At this time, it would make sense to allow other connections to get the available bandwidth. Indeed, the XQM agent, through its buffer matching mechanism for efficiency control, allows connections AB to grab the available bandwidth without hurting overlay connections AD.

Experiment 3: The previous experiment illustrated that for a particular connection to get its fair-share, it may not be enough for the XQM agent at the OTM node to simply *"not introduce additional losses"*. In particular, when exogenous loss rates are fairly high, some losses should be hidden from the sender. We propose (and present results of) a technique that enables OTMs to mark packets so network underlay routers would not drop these packets (e.g., using Assured Forwarding settings in DiffServ). Other techniques, such as forward error correction, can also be employed at the overlay traffic managers to "hide" excessive exogenous losses.

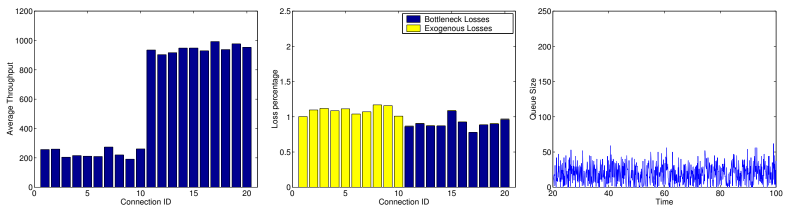
The plots in rows (d) and (e) of Figure 6 show two different ways of implementing exogenous loss hiding in the XQM agent. In the first, once $q(t)$ reaches zero (implying that the XQM agent need not introduce any additional losses), we trigger hiding as well whereby we mark all packets so they won't get dropped at underlay nodes. Once the $q(t)$ goes above zero, hiding is stopped. A drawback of this scheme is the potential for oscillations due to the alternation in control rules. Nonetheless, this scheme is able to improve the fairness as well as maintaining the queue size at the target level. The plots in row (f) of Figure 6 provide a remedy for this, whereby the XQM agent smoothly tunes the level of hiding (i.e., incrementally increasing it) when $q(t)$ is negative. This technique is not susceptible to the negative impacts from a sudden alteration in control rules. Overlay connections AD experience only the quiescent amount of exogenous losses, allowing them to reach their fair shares.



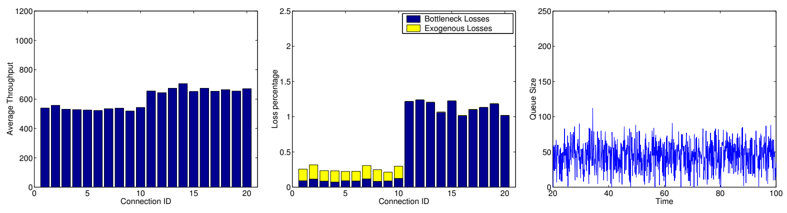
(a) RED [Fairness Index = 0.63]



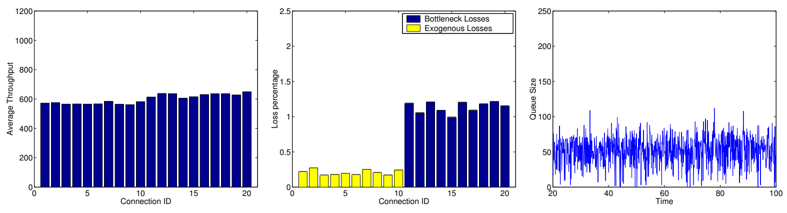
(b) FRED [Fairness Index = 0.69]



(c) XQM PI-TB [Fairness Index = 0.73]



(d) XQM PI-TB (100% hiding when $q(t) \leq 0$ and disabled otherwise)
[Fairness Index = 0.98]



(e) XQM PI-TB (smooth hiding as a function of $q(t)$, when $q(t) \leq 0$)
[Fairness Index = 0.99]

Fig. 6. Comparative performance of various AQMs, showing average throughput (left), overall loss rate (middle), and instantaneous buffer size (right).

6.2 XQM Design Extensions

An XQM agent at an overlay traffic management node can easily achieve any weighted allocations of throughput as equation (15) suggests. We just need to assign to the target connection's rate $\hat{x}_i$, the weighted fair share instead of an equal fair share. This means that the XQM agent can remove the bias between connections with different RTTs. It can also deal with flows as aggregates. Aggregation would be based on flows that have similar characteristics. We demonstrate this in [22]. Although in this paper, we focused on long-lived TCP flows to clarify fairness issues, an XQM agent would deal with UDP flows in the same way. Our results in a mixed TCP/UDP Web/FTP/CBR environment confirm the superiority of XQM in the presence of exogenous losses.

7 Conclusion

In this paper, we captured the effect of exogenous packet losses by extending a dynamic fluid model of TCP. As one would expect, we showed that high levels of exogenous losses lead to inefficiencies. Surprisingly though, we also found that low levels of exogenous losses that don't force TCP below its fair share, improve fairness among flows. We argue that exogenous-loss awareness should be taken into account in the design of overlay traffic managers (OTMs) that aim to achieve global fairness. Indeed, we showed that the road to global fairness requires accounting for exogenous losses and, accordingly, invoking the right control rules at the right time scale. We proposed an eXogenous-loss aware Queue Management (XQM) approach for use in OTMs. XQM promotes fairness without compromising efficiency. In contrast to AQM-like designs, an XQM agent uses exogenous losses as *carriers* of its own feedback signal, hiding such losses *only* when they reach levels that jeopardize global fairness, and *only* to the extent necessary to avoid such unfairness.

References

[1] A. Bakre and B. Badrinath, I-TCP: Indirect TCP for Mobile Hosts, in: Proceedings of the 15th ICDCS, 1995.

[2] H. Balakrishnan, S. Seshan and R. Kartz, Improving Reliable Transport and Handoff Performance in Cellular Wireless Networks, ACM Wireless Networks, 1(4).

[3] C. Jin, D. Wei and S. Low, Internet Draft: FAST TCP for high-speed long-distance networks , draft-jwl-tcp-fast-01.txt.

[4] D. Katabi, M. Handley and C. Rohrs, Internet Congestion Control for High Bandwidth-Delay Product Networks, in: Proceedings of ACM SIGCOMM, San Diego, CA, 2002.

[5] S. Floyd, Internet Draft: HighSpeed TCP for Large Congestion Windows, draft-ietf-tsvwg-highspeed-01.txt.

[6] S. Biaz and N. Vaidya, Distinguishing Congestion Losses from Wireless Losses using Inter-Arrival Times at the Receiver, in: IEEE ASSET 1999, Richardson, TX, 1999.

[7] C. Parsa and J. Garcia-Luna-Aceves, "Differentiating Congestion vs. Random Loss: A Method for Improving TCP Performance over Wireless Links", in: WCNC 2000:IEEE Wireless Communications and Networking Conference, Chicago, IL, 2000, pp. 23–28.

[8] T. Kim, S. Lu and V. Bharghavan, "Improving Congestion Control Performance Through Loss Differentiation", in: ICCCN 1999, Boston, MA, 1999.

[9] D. Barman and I. Matta, Effectiveness of Loss Labeling in Improving TCP Performance in Wired/Wireless Networks, in: IEEE ICNP'2002, Paris, France, 2002.

[10] R. Puri, K. Lee, K. Ramchandran and V. Bharghavan, An Integrated Source Transcoding and Congestion Control Paradigm for Video Streaming in the Internet, IEEE Transactions on Multimedia 3 (1).

[11] S. Floyd and V. Jacobson, Random Early Detection Gateways for Congestion Avoidance, IEEE/ACM Transactions on Networking, 1993.

[12] S. Low, F. Paganini, J. Wang, S. Adlakha and J. Doyle, Dynamics of TCP/RED and a Scalable Control, in: Proceedings of IEEE INFOCOM 2002, New York, NY, 2002.

[13] C. Hollot, V. Misra, D. Towsley and W. Gong, A Control Theoretic Analysis of RED, in: Proceedings of IEEE INFOCOM, 2001.

[14] V. Jacobson, Congestion Avoidance and Control, in: Proceedings of ACM SIGCOMM'88, Stanford, CA, 1988.

[15] L. Subramanian, I. Stoica, H. Balakrishnan and R. Katz, OverQoS: An Overlay Based Architecture for Enhancing Internet QoS, in: Proceedings of NSDI, 2004.

[16] M. Guirguis, A. Bestavros, I. Matta, N. Riga, G. Diamant and Y. Zhang, Providing Soft Bandwidth Guarantees Using Elastic TCP-based Tunnels, in: Proceedings of the Ninth IEEE Symposium on Computers and Communications (ISCC), Alexandria, Egypt, 2004.

[17] G. Diamant, L. Veytser, I. Matta, A. Bestavros, M. Guirguis, L. Guo, Y. Zhang and S. Chen, itmBench: Generalized API for Internet Traffic Managers, in: Proceedings of the 10th IEEE Globecom Workshop on Computer-Aided Modeling, Analysis and Design of Communication Links and Networks, Dallas, TX, 2004.

[18] S. Athuraliya, S. Low, V. Li and Q. Yin, REM: Active Queue Management , IEEE Network 15 (3) (2001) 48–53.

[19] F. Kelly, Mathematical Modelling of the Internet, Mathematics Unlimited - 2001 and Beyond (2001) 685–702.

[20] S. Shenker, A Theoretical Analysis of Feedback Flow Control, in: Proceedings ACM SIGCOMM'90, Philadelphia, PA, 1990.

[21] E. A. et al., UCB/LBNL/VINT Network Simulator - ns (version 2).

[22] M. Guirguis, Exogenous-Loss Aware Traffic Management in Overlay Networks: Toward Global Fairness, MA Thesis, Boston University. URL
`http://cs-people.bu.edu/msg/research/papers/MA-Thesis.pdf`

[23] The PSC Networking group, The TCP-Friendly Website, http://www.psc.edu/networking/tcp_friendly.html.

[24] D. Chiu and R. Jain, Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks, Computer Networks and ISDN Systems 17 (1989) 1–14.

[25] T. Ott, T. Lakshman and L. Wong, SRED: Stabilized RED, in: Proceedings of INFOCOM 1999, New York, USA, 1999.

[26] C. Hollot, V. Misra, D. Towsley, W. Gong, On Designing Improved Controllers for AQM Routers Supporting TCP Flows, in: Proceedings of IEEE INFOCOM 2001, Anchorage, AL, 2001.

[27] D. Lin, R. Morris, Dynamics of Random Early Detection, in: Proceedings of ACM SIGCOMM'97, Cannes, France, 1997.

[28] N. F. Maxemchuck, Dispersity Routing in High-Speed Networks,, Computer Networks and ISDN Systems 25 (1993) 645–661.

[29] S. Jaiswal, G. Iannaccone, C. Diot, J. Kurose, D. Towsley, Measurement and Classification of Out-of-Sequence Packets in a Tier-1 IP Backbone, in: IEEE INFOCOM 2003, San Francisco, CA, 2003.

[30] T. Bu, D. Towsley, Fixed Point Approximations for TCP Behavior in an AQM Network, in: ACM SIGMETRICS, Boston, MA, 2001.

[31] S. Low, L. Peterson, L. Wang, Understanding TCP Vegas: A Duality Model, in: ACM SIGMETRICS 2001, Boston, MA, 2001.

[32] F. Kelly, A. Maulloo, D. Tan, Rate Control for Communication Networks: Shadow Prices, Proportional Fairness and Stability, Journal of Operations Research Society.

[33] R. Gibbens, F. Kelly, Resource Pricing and the Evolution of Congestion Control, Automatica 35 (1999) 1969–1985.

[34] S. Low, D. Lapsley, Optimization Flow Control, I: Basic Algorithm and Convergence, IEEE/ACM Transactions on Networking, 1999.

[35] S. Kunniyur, R. Srikant, Analysis and Design of an Adaptive Virtual Queue (AVQ) Algorithm for Active Queue Management, in: Proceedings of ACM SIGCOMM 2001, 2001, pp. 123–134.

[36] R. J. Gibbens, F. P. Kelly, Distributed Connection Acceptance Control for a Connectionless Network, in: Proceedings of 16th International Teletraffic Congress, 1999, pp. 941–952.

[37] A. Demers, S. Keshav, S. Shenker, Analysis and simulation of a fair queueing algorithm, Internetworking: Research and Experience, 1990.

[38] A. Parekh, R. Gallager, A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: the Single node case, IEEE/ACM Transactions on Networking, 1993.