Congestion Control and Periodic Behavior

Anna C. Gilbert, Youngmi Joo, Nick McKeown
AT&T Labs-Research, 180 Park Avenue, Florham Park, NJ 07932-0971
Stanford University, Stanford, CA 94305

***Abstract* -- Feedback based congestion control — used by TCP and other transport protocols — causes the transmission rate of a long-lived flow to oscillate. This paper is about the tendency of multiplexed flows on a bottleneck link to oscillate causing “aggregate periodic behavior”. As a consequence, bottleneck links can become underutilized, wasting capacity. One simple way to prevent this is to use larger buffers in the bottleneck router. We explore how large these buffers need to be to absorb oscillations. Another proposed solution is to use a randomized active queue management algorithm, such as RED. A goal of RED is to reduce “global synchronization” of TCP flows, and hence avoid oscillations. Our results suggest that RED does not reduce aggregate periodic behavior when compared to Drop Tail. Perhaps surprisingly, our results and a simple analytical model suggest that the tendency for aggregate traffic to oscillate is robust against many forms of disturbances added to the timing of feedback and reaction, even if each constituent flow shows little or no sign of periodicity. Different flows can be in different states and yet conspire to make their sum have periodic behavior.**

I. INTRODUCTION

One reason for the success of TCP and its widespread usage in the Internet is its ability to control network congestion.

A TCP source uses implicit notification of network congestion (via packet loss) to control the rate at which it sends data. The rate is controlled by increasing or decreasing the window of outstanding, unacknowledged data. In particular, TCP uses “additive increase and multiplicative decrease” to increase the rate linearly during times of no packet loss, and to decrease the rate geometrically when loss occurs. As a consequence, the rate at which a TCP source transmits tends to be periodic over time. For example, Figure 1 shows the periodic behavior of a single TCP source in an otherwise idle network.

When many TCP sources compete for the buffers in a congested router, packet loss causes each individual flow to exhibit periodic behavior. A question worth asking is: Is the *aggregate* behavior of all of the flows also periodic? For example, consider Figure 2(a) which shows a

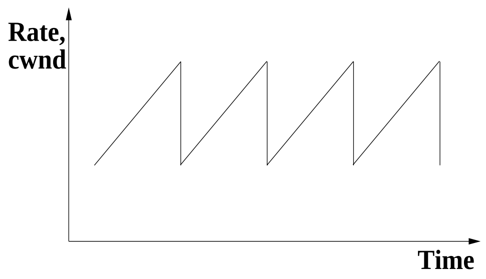


Figure 1: Periodic oscillation shown by a single TCP connection

number of periodic TCP flows. If these flows were to pass through a single router over a shared link, will the aggregate flow be smooth (as in Figure 2(b)) or periodic (as in Figure 2(c))?

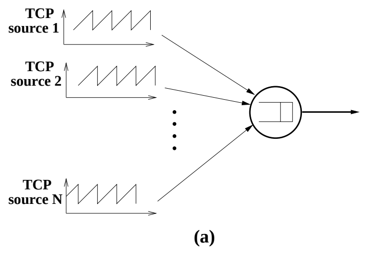
It has been previously observed via simulation [7][8][9] that the aggregate packet arrival rate at the bottleneck is periodic.

So why should we care if the aggregate rate of traffic on a link is periodic? Because the average of any periodic signal is lower than its peak, which means that a link with periodic traffic is under-utilized. For example, the average data rate in the output link of Figure 2(c) is significantly lower than that of Figure 2(b).

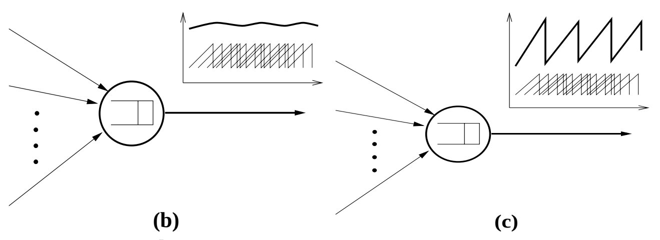
Many experiments[7] and the intuitive explanations of these experiments suggest that TCP sources competing for bandwidth on a congested link will synchronize through the weak coupling inherent in congestion control. Intuitively speaking, a population of sources will synchronize because they all experience loss at roughly the same time, they all scale back their transmission rate in the presence of these losses, and then they increase their transmission rate until the next bout of congestion occurs.

In order to understand if, and by how much, periodic behavior affects the performance of individual TCP flows, we need to examine the bottleneck link. If the bottleneck link is under-utilized then all flows suffer reduced throughput.

Two possible ways to reduce periodicity on a bottleneck link are: (1) Increase the size of the buffers immediately prior to the link in order to absorb the oscillations and filter them out, and (2) Use a randomized drop-pol-



(a)



(b) (c)

Figure 2: Multiple TCP connections; (a)topology, (b) flat aggregate rate, (c) periodically oscillating aggregate rate

icy in the routers. In particular, it is a stated goal of RED [4] to *desynchronize* each of the flows sharing a congested router buffer so that the periodic behavior of each individual flow is not synchronized with the others. RED discards packets randomly in an attempt to cause different TCP sources to reduce (and hence increase) their rates at different times. Intuition suggests that if the flows are not oscillating in phase, then their aggregate behavior will be smooth (like in Figure 2(b)) and hence link utilization will be high.

In this paper, using simulation and simple topologies we set out to explore: (1) How much utilization is lost due to periodic behavior, (2) How much buffering should be deployed by network designers in order to filter out the periodicity, and (3) How effective is RED in reducing or eliminating periodicity.

Our simulation results (which are based on simple topologies) suggest that if buffers are too small, periodic behavior occurs on bottleneck links, both with and without RED, even if sources are distributed at random distances from the bottleneck. RED does not appear to reduce periodicity and can, under certain circumstances, increase it. Perhaps the most surprising finding of this paper is that simulations suggest that the tendency for aggregate traffic to oscillate is quite strong, even if each constituent flow shows little or no sign of periodicity.

In an attempt to capture and characterize the way that multiple seemingly non-periodic flows might interact to cause aggregate periodic behavior, we developed a simple analytical "toy" model of TCP's congestion control algorithm. Although too simple to draw strong conclusions from, the model tends to support the simulation results that there is a strong tendency for traffic to oscillate, and that it is likely to be a characteristic of almost any feedback congestion-control mechanism in which the buffers are (too) small.

II. Simulation Results

A. Simulation Setup

We use ns-2[1][2] for our simulations, with TCP Reno sources and the topology shown in Figure 3. Fifty clients (at the nodes $D_1, D_2 \ldots D_N$, with $N = 50$) on one side of a bottlenecked router retrieve an infinite-length file from a server, S, on the other side. Hence, there are 50 TCP connections, sharing a single oversubscribed 1.5 Mbps link between the nodes M_2 and M_3. Each connection starts at some random time between 0 and 300 seconds, lasting the duration of the simulation (4200 seconds). Figure 3 shows the data rate and propagation delays for each link. In our simulations, we choose the propagation delays between each endpoint D_i, $i = 1, 2, \ldots, 50$ and the node M_3 to be either a single value of 20 msec (which we call "fixed RTT"), or pick them at random and uniformly over the range 0 to 1 seconds (which we call "variable RTT").

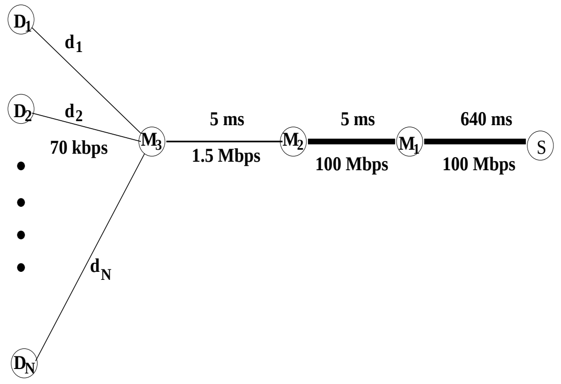


Figure 3: Simulation Setup

We compare both Drop Tail and RED queue management disciplines, with varying buffer sizes and various threshold values (min_{th} and max_{th}) for RED.[1] When varying min_{th} and max_{th} we use two different ways. One is to fix min_{th} to a very small value (equal to five), while varying the max_{th} values. We call this "RED minfix" throughout the paper. Another is to vary both min_{th} and max_{th}, while keeping the ratio of max_{th} to min_{th} a constant value of three. We call this "RED ratiofix". The purpose of each is to see whether the phenomena observed with RED have more to do with the difference

1. Other parameters for RED used in simulations are in Appendix B.

between the two threshold values, or the combined effect of the difference and the value of min_{th}.

B. *Periodic Behavior with Small Buffers (Impact of Insufficient Buffer Space)*

Our first results show the effect of small buffers in the router just prior to a bottleneck link. Expecting that small buffers will exaggerate any periodic behavior, we explore how much utilization is lost, if any. Specifically, we keep the total buffer space, or max_{th} of RED, at around 20% of the "bandwidth-RTT product".

Figure 4 shows the simulation results for DropTail (Figure 4(a), (b)) and RED ((c), (d)), with fixed RTT of 1340 ms (Figure 4(a),(c)) and variable RTT (Figure 4(b),(d)). Periodic behavior is clearly visible in all four graphs, which leads us to the definition of wasted utilization on the bottleneck link:

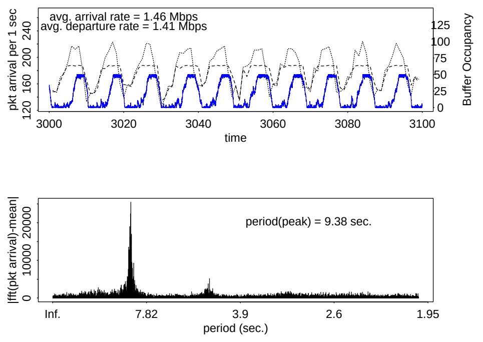
$$\text{loss of utilization} = 100 \cdot \left(1 - \frac{\text{average departure rate}}{\text{line rate}}\right).$$

Drop Tail in Figure 4(a)(b), shows quite noticeable periodicity, leading to a loss in utilization of approximately 5-7%. This is to be expected — Drop Tail has been previously described as leading to "global synchronization".

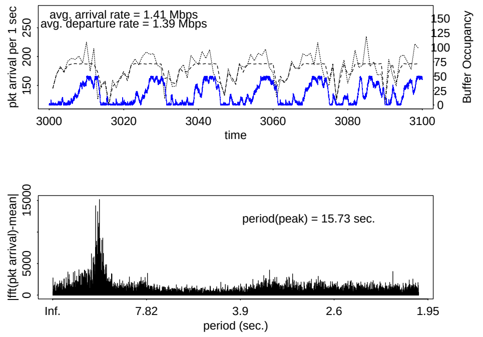
Our simulation results for RED in Figure 4(c)(d) challenge both the term "global synchronization" and the efficacy of RED in preventing it, at least when the buffer size is small. In this example, the loss in utilization with RED is approximately 15% — larger than for Drop Tail.

We shall see that "global synchronization" is not really a good characterization of the problem for Drop Tail.

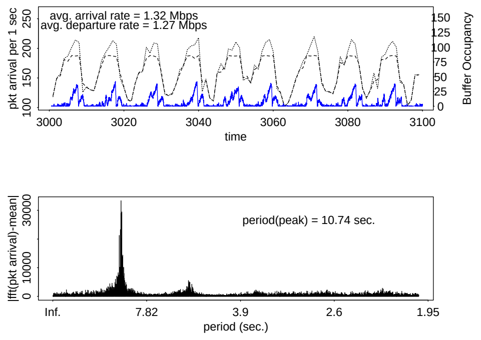
We start here by testing the intuition that periodic behavior comes from synchronized sources. Figure 5 shows that it does not. (We'll be spending quite a bit of time in the remainder of this paper trying to understand how periodic aggregate behavior occurs even when each individual flow shows little sign of oscillation.) The figure shows the total `wnd`[2] values plotted against time, along with `cwnd` values for 3 randomly chosen TCP flows. `cwnd` represents an individual TCP flow's congestion 'state', while the 'total `wnd`' represents the aggregate state of flows on the link.[3] The top half of Figure 5 shows total `wnd` oscillating in a sawtooth pattern, in agreement with the oscillatory behavior of the aggregate packet arrival rate. However, individual `cwnd` val-



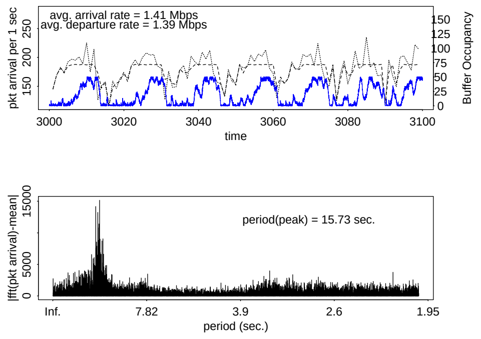
(a)



(b)



(c)



(d)

Figure 4: Aggregate Packet Arrival, Departure rates and queue lengths for (a) DropTail, fixed RTT, (b) DropTail, variable RTT, (c) RED, fixed RTT, (d) RED, variable RTT

[2]. Sum of `wnd`=min(`cwnd`, `rwnd`) values for all flows.

[3]. It has been also checked from simulation results that the total `wnd` values are roughly the same as the packet arrival rate on the time granularity of RTT+queueing delay on the path.

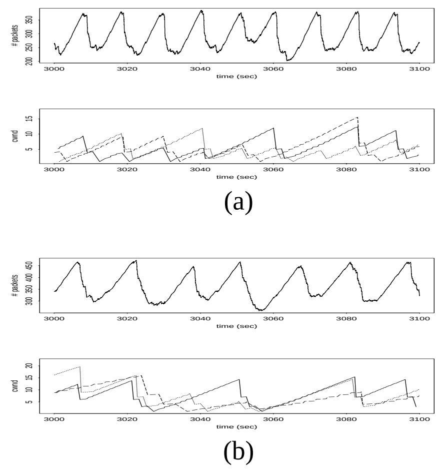


Figure 5: total `wnd` values (top) and `cwnd` values of three randomly chosen flows, (a) fixed RTT (1340ms), (b) variable RTT.

ues are not necessarily in sync with one another, as shown in the bottom half of Figure 5. Instead, at each congestion epoch some `cwnd` values continue to increase while others do not. Note that, although Figure 5 (a) is consistent with the results in [10] which indicates that the individual `cwnd` behavior of the sources appears chaotic, simulation results show that the aggregate behavior is decidedly deterministic.

Surprisingly, introducing randomness in packet drops doesn't change the aggregate periodic behavior, neither at the input nor at the output of the buffer. The outgoing line utilization is lower in RED in Figure 2, when compared with Drop Tail.

We note that RED has little room to exercise randomness when the buffers are too small. Since $max_{th} - min_{th}$ is small, the majority of the packet drops occur in bursts that are not much different from Drop Tail. Moreover, since RED drops packets until the *time-averaged queue length* falls below both of its thresholds, RED tends to punish flows more severely than Drop Tail. This becomes clear when we look at how many flows experienced multiple packet drops[4] within a burst, or the number of drops when packet drops come in well-separated bursts. Figure 6 shows histograms of those counts for simulation results shown in Figure 4 (a) and (c), observed at each congestion epoch. Note that RED forces more flows to reduce their `cwnd` (and hence transmission rate) compared to Drop Tail. Consequently the aggregate arrival rate decreases more with RED, making the buffer empty for longer periods of time. Also, since the maximum buffer occupancy is smaller for RED than Drop Tail, the queues will go empty quicker for RED once the aggregate arrival rate falls below the outgoing line rate.

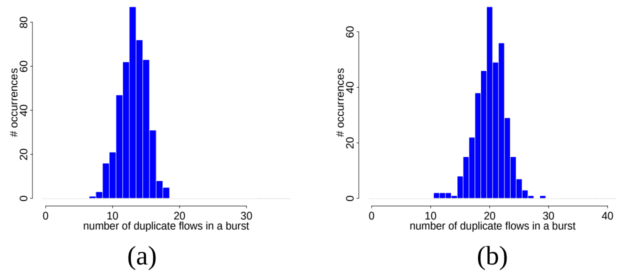


Figure 6: Histogram of number of flows with multiple packet drops within a burst, (a) DropTail(Figure 4 (a)), (b) RED (Figure 4 (c)).

C. *Periodic Behavior with Large Buffers*

We can expect that when the router buffer is large enough, the bottleneck link will no longer exhibit periodic behavior, due to the smoothing effect of the buffer. Simulations indeed show this to be the case. While the links immediately prior to the buffer continue to oscillate, the output link from the buffer does not regardless of the queue management algorithm.

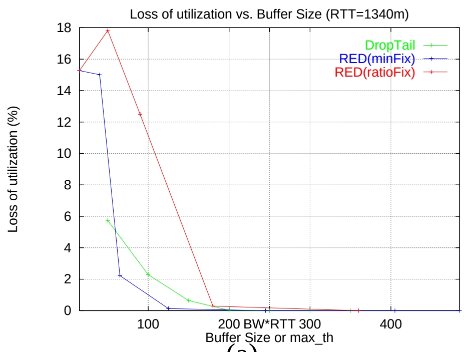
It is of practical interest to know how large the buffer needs to be to prevent periodic behavior from occurring for both Drop Tail and RED. Armed with some data, network operators may be able to prevent oscillations from occurring and hence increase link utilization. In what follows, we provide quantitative answers on buffer sizing for the network topology considered in our simulations.

1) *Preventing Periodic Behavior*

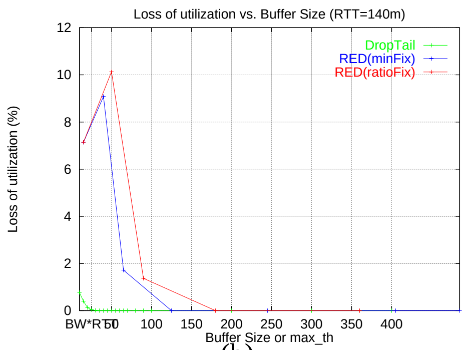
First we'll try and prevent periodic behavior with Drop Tail. We tested the well-known rule-of-thumb that the buffers at the bottleneck router should be at least equal in size to the bandwidth-RTT product. Note, however, that this value comes from assuming that the aggregate arrival pattern is a 'pulse train', i.e. it consists of bursts of packet arriving (exactly) every RTT, which we know from our simulation results is not true. Despite the difference in behavior, we found that with Drop Tail and one bandwidth-RTT of buffering, the bottleneck link was 100% utilized for both fixed and variable RTT (for which we substituted the average RTT of 2300ms). The loss of utilization as a function of buffer size is shown in Figure 7 for both incoming[5] and outgoing links.

For RED, it is less clear how to interpret the band-

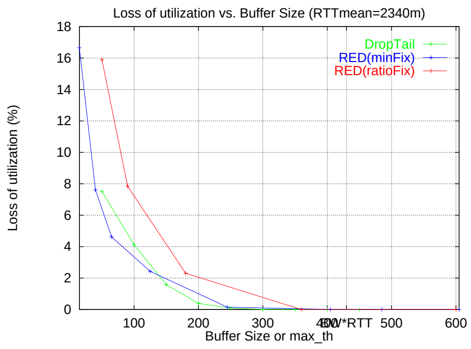
[4]. We regard this as a congestion notification with a higher magnitude, since TCP Reno source is likely to reduce `cwnd` to 1 when this happens [3]. Therefore, this particular interpretation is closely tied to the details of the TCP implementation.



(a)



(b)



(c)

Figure 7: Loss of utilization on ingress and egress links, (a)long fixed RTT, (b) short fixed RTT, (c) variable RTT.

width-RTT guideline: Is it only the max_{th} value that matters, or should the bandwidth-RTT product lie somewhere between min_{th} and max_{th}?

So we considered various values of min_{th} or max_{th} (RED minFix, RED ratiofix). In each case, the loss of utilization eventually falls to zero, as max_{th} increases. A

[5] We define the loss of utilization on the ingress link as $100(1 - (\text{Avg Arrival Rate}/\text{Egress Line Rate}))$. Since the average arrival rate can be larger than the outgoing line rate, this value can be negative, which indicates that packets are being buffered. A positive value indicates that the arrival process, even without considering the buffer becoming empty, is 'inefficient' due to its periodic behavior.

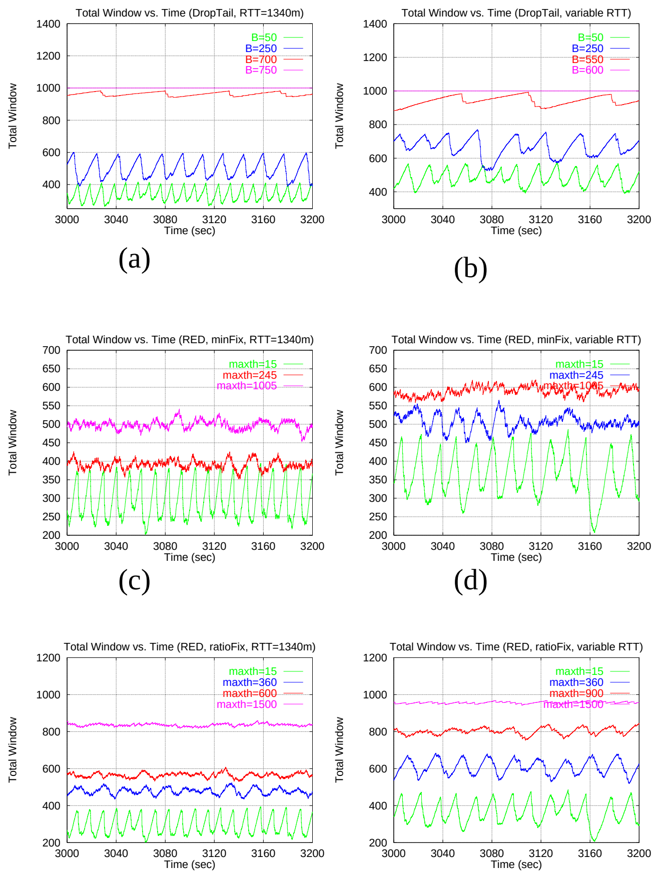


Figure 8: total wnd values vs. time for different buffer spaces, (a)DropTail, fixed RTT, (b) DropTail, variable RTT, (c) RED minfix, fixed RTT, (d) RED minfix, variable RTT, (e) RED ratiofix, fixed RTT, (f) RED ratiofix, variable RTT

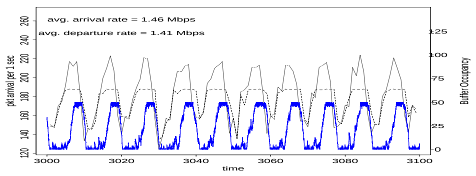
few things to note are:

- Comparing RED minfix and RED ratiofix, it appears that max_{th} has more impact on the loss of utilization. In Figures 5(a)-(c), RED minfix needs smaller max_{th} than ratiofix to achieve 100% utilization, when the two have roughly the same $max_{th} - min_{th}$.
- When RTT is short compared to the transmission time of one wnd of data, as in Figure 7 (b), RED requires much larger buffer space than DropTail to achieve 100% utilization.

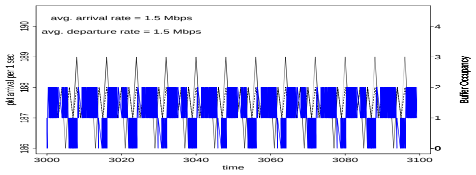
2) Changes in Aggregate Behavior with Increasing Buffer Space

In this subsection, we examine how the *collective* behavior of the sources changes as buffer size and thresholds increase.

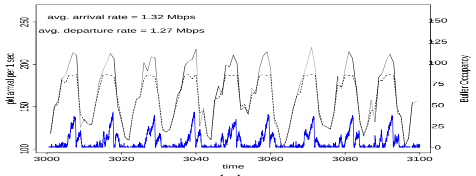
Figure 8 shows total wnd values as a function of time for different buffer sizes. For Drop Tail, as buffer size increases, total wnd stays constant. This corresponds to all 50 flows transmitting with their wnd values set to the



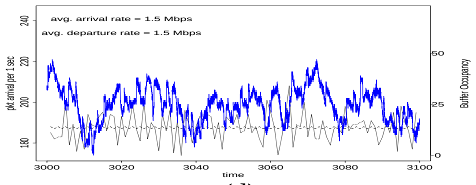
(a)



(b)



(c)



(d)

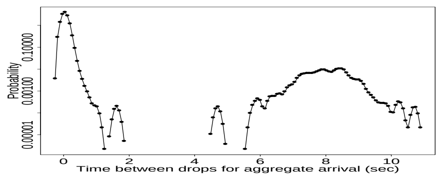
Figure 9: Aggregate packet arrival (a) DropTail with buffer size of 50, (b) DropTail with buffer size 750, (c) RED with maxth=15, (d) RED with maxth=1500.

RED shows different trends in total `wnd` values as max_{th} value increases for both ‘minfix’ and ‘ratiofix’ cases; the magnitude of oscillation (of total `wnd`) decreases, eventually staying constant. But the overall values of total `wnd` do not increase as rapidly as in Drop Tail, although the consequences of this “convergence” leads to the elimination of periodicity in the aggregate packet arrival rate. This is verified in Figure 9, which shows the packet arrival and departure rates for both Drop Tail and RED ‘minfix’ with varying buffer size (for Drop Tail) and threshold values of $max_{th} = 15$ and $max_{th} = 1500$ (for RED). Although the queue length still fluctuates with $max_{th} = 1500$, there isn’t any prominent periodic fluctuation in the aggregate arrival process.

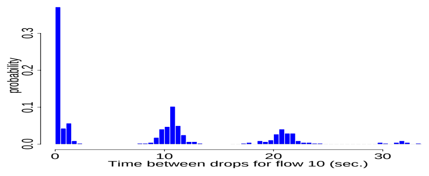
Disappearing fluctuations of total `wnd` values as max_{th} increases can be explained by changes in timing dynamics of packet drops. As max_{th} increases past the bandwidth-RTT product and beyond, RED begins to have enough opportunities to introduce randomness in packet drops before the time-averaged queue length reaches maximum possible value, and the buffer size appears to be large enough to accommodate this. We can verify this through a simple calculation. If the buffer size is large enough to satisfy the equality

$$\frac{(\text{Number of flows} \times \text{Maximum cwnd})}{(\text{Bottleneck Bandwidth})}$$
$$= \text{RTT} + \text{Maximum Queueing Delay}$$

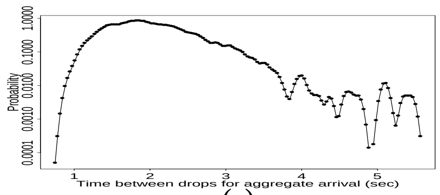
where, for Drop Tail, the maximum queueing delay is roughly (buffer size)/(bottleneck bandwidth), then the network can accommodate all the sources transmitting at their peak rates without dropping any packets. In one simulation shown in Figure 7(a) we had RTT(without queueing delay) of 1340 ms, bottleneck bandwidth 1.5 Mbps, number of flows 50, maximum `wnd(=rwnd)` 20 packets, and the packet size of 1000 bytes. This yields the buffer size of 748 packets, which is in agreement with the buffer size of 750 packets that was just able to completely eliminate any aggregate periodic fluctuation for Drop Tail.



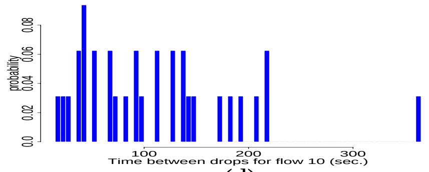
(a)



(b)



(c)



(d)

Figure 10: Distribution of inter-drop times of packet, (a) max_th=15, aggregate, (b) max_th=15, each flow, (c) max_th=1500, aggregate, (d) max_th=1500, each flow.

max_{th}, beyond which point it drops packets deterministically. As a result, packet drops at the bottleneck no longer occur in short, clearly delineated bursts, occurring instead at random intervals. The intervals are typically longer than the back-to-back inter-drop times, and generally increase with max_{th}, as shown on the left half of Figure 10. Figure 10 (a) and (b) are for $max_{th} = 15$ and (c) and (d) for $max_{th} = 1500$. For the larger value of max_{th}, the distribution of inter-drop time is centered around 2.0 second, whereas the distribution for the smaller max_{th} has two distinctive 'islands', one for the back-to-back packet drops, the other for the packet drops belonging to neighboring 'bursts'.

The same trend is observed when we look at inter-drop times seen by each individual flow, as shown in the right side of Figure 10, at the top and bottom, respectively. An important consequence is that the flows are no longer all driven in to the same state (namely, `cwnd`=1, as a result of a burst of packet drops), hindering or eliminating the aggregate periodic behavior.

III. A "Toy" Analytical Model

Simulation has only limited value — it can only tell us definitive things about very specific topologies. While we may convince ourselves that our experiments are representative, we cannot, in any scientific way, conclude that our results extend to more general network or situations.

So, in an attempt to capture and characterize the way that a feedback-based congestion control algorithm might cause aggregate periodic behavior, we created a very simple analytical model. The model attempts to crudely capture the feedback behavior of TCP with a random drop policy such as RED. Our analysis suggests that although sources may be placed at different distances from the bottleneck, and that they may initially start asynchronously to each other, the feedback congestion control algorithm will eventually cause the aggregate behavior to oscillate. Perhaps surprisingly, the model suggests that the oscillation is quite stable in that it remains prominent in spite of various irregularities and perturbations. It appears that it would take quite a drastic change to the feedback or drop mechanism to prevent the periodic behavior.

The conclusions derived from this model are valid only in the regime where buffering is inconsequential, when the buffer is suitably small. The analytical results show that the aggregate periodic behavior is an extremely stable feature of a small buffer network configuration. However, the model does not accurately predict the departure process from a large buffer, a situation in which we in simulations can eliminate oscillatory behavior. A more complete model would have to take such buffering into account for a more accurate picture.

A. Description of model

We model a TCP connection as a source with an infinitely long file to transmit to a receiver across a bottleneck link. There are N sources or connections sharing the bottleneck link which has capacity C. In the absence of congestion or feedback from the bottleneck, each source increases its sending rate linearly over time (i.e., $\frac{d}{dt}x_i(t) = 1$ where $x_i(t)$ is the rate of source i at time t). The system of sources experiences congestion when the total rate $A(t) = \sum_{i=1}^{N} x_i(t)$ reaches the link capacity C. At each congestion epoch when $A(t) = C$, a fraction r of the N sources (chosen uniformly at random) experiences losses and these r sources receive a notification of congestion. Each source upon congestion notification, resets its transmission rate to zero. We assume that these events signaling congestion and resetting the transmission rate occur instantaneously when the aggregate rate reaches the link capacity. After reset, the rates of the sources continue to grow linearly until the system reaches the next congestion epoch, at which point the process repeats with another random fraction r of sources experiencing losses and adjusting their rates. This model assumes that there is an entity at the bottleneck link that can measure the instantaneous aggregate rate, signal, and reset the sources instantaneously.

B. Conclusions drawn

Rather than following the possible states of all N sources, we assume that N is large enough for us to employ a mean-field approximation for the system. We describe the state of the system at time t by a density $\rho(x, t)$ that represents the percentage of sources with transmission rate x at time t. We let $\rho(x, t)$ take on any real positive value, not necessarily an integer multiple of $1/N$. Furthermore, we approximate the result of each reset upon $\rho(x, t)$ as removing a uniform percentage r of the sources at rate x and time t (i.e., that the approximate effect upon the population of sources is the mean or expected effect) and resetting the rates of that percentage r by placing a point mass of weight r at rate $x = 0$. If $\rho(x, t^-)$ represents the distribution of sources immediately before the nth congestion epoch, then the mean-field approximation gives us the representation

$$\rho(x, t_n^+) = r \cdot \delta_0(x) + (1 - r) \cdot \rho(x, t_n^-)$$

for the distribution of source rates immediately following the reset of the sources. In addition, the linear increase of the sources' rates between congestion epochs tells us that the configuration $\rho(x, t + \delta t)$ of the system at time $t + \delta t$ is simply the translation of the distribution at time t by δt, $\rho(x, t + \delta t) = \rho(x - \delta t)$.

We demonstrate that:

- there is a configuration ρ_* of the source rates that is invariant under the reset operation at each congestion epoch; i.e., the distribution of source rates immediately following a reset is the same as the distribution after the preceding congestion epoch,
- there is a constant time interval $\Delta t = (r \cdot C) / N$ between congestion epochs,
- the aggregate rate of the sources $A(t)$ is a Δt-periodic function that oscillates between a maximum value C and a minimum value $(1 - r) \cdot C$ with an average of $C - r \cdot C / 2$, and
- any initial configuration of the sources will tend to this invariant exponentially fast in r.

The reader interested in technical details will find them in Appendix A. See Figure 11 for a picture of this configuration, where the mean field approximation is plotted in black for $N = 50$ connections and $r = 1/4$.

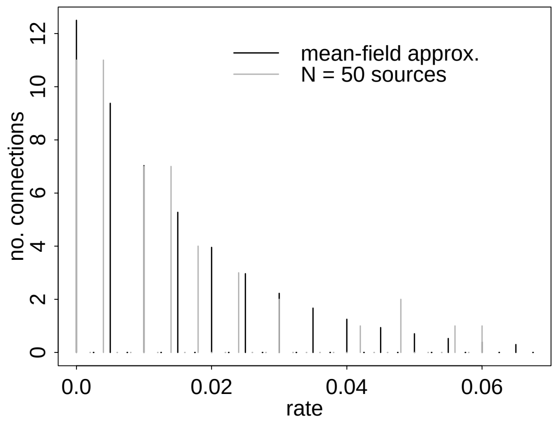


Figure 11: The invariant configuration of sources for the mean-field approximation and a simulated instance of the model with N = 50 sources, the fraction r = 1/4 of the sources reset randomly at each epoch, the bottleneck link capacity C=1, and the epoch duration $\Delta t = r/N = 0.002$. The distribution of 50 sources after 1000 cycles in a realization of the toy model is also plotted in grey.

The invariant distribution of source rates has two important properties. First, it consists of distinct populations of sources, each population with one rate, rather than constraining all N sources to transmit at the same rate. These distinct populations point out a slight misnomer in the term "synchronization" commonly used to describe this phenomenon. The sources do not synchronize their rates with respect to one another, they conspire in groups to produce an aggregate periodic rate, and so the term "aggregate periodic behavior" is a more accurate description of this phenomenon. The sizes of the rate populations do decay exponentially, showing that many of the sources transmit at the lowest rate while very few sources transmit at a high rate.

The second important property the invariant configuration possesses is stability. To explore the stability of the invariant distribution, we define a perturbation of the system at the ith congestion epoch to be a random deviation from the time the ith reset occurs. We choose ε_i uniformly at random in $[-\varepsilon, \varepsilon]$ and jitter the duration of the ith epoch $\Delta t_i = \Delta t + \varepsilon_i / N$. The time Δt between the $(i - 1)$st reset and when the sources next reach link capacity C is still $(r \cdot C) / N$, we simply randomly perturb when this reset occurs as we do in a queueing policy that attempts to inject a small amount of randomness into the loss process. At the new time t_i, we reset a uniformly chosen random fraction r of the sources and let the system evolve again until the total rate exceeds capacity C. We repeat this process for each congestion epoch (drawing an independent ε each time) and show that the perturbed system remains close to the unperturbed system (which, we emphasize, is converging exponentially fast to the invariant distribution independent of the initial conditions). The perturbed system differs from its original trajectory with respect to its rates by a maximum value of ε and with respect to its congestion epoch durations by ε / N. On average, the perturbed system does not stray from its original course, as the perturbation ε is a random variable with mean 0.

C. *Extensions and generalizations*

Several aspects of this mathematical model are only simple abstractions of more complex realistic behavior. To generate a less artificial picture of the behavior of a system of N sources, we must change both the manner in which the sources increase their rates and the rate reset policy in the face of congestion. However, neither modification changes the gross behavior of the system and so we feel reasonably confident drawing conclusions from the behavior of the simple abstract model.

The rate increase profile in this simple mathematical model does not reflect the complex behavior of the rate increase in TCP with its slow-start and congestion avoidance phases. However, if we impose a more complicated rate increase $\frac{d}{dt} x_i(t)$, we do not change the overall behavior of the sources although we do change the details. The congestion epoch duration is different and is given by

$$A(t^-_{n+1}) = C = (1-r) \cdot C + N \cdot \int_{t_n}^{t_{n+1}} \frac{d}{dt} x_i(s) ds$$

but with the same reset policy, the system evolves as before. The source rate distribution moves according to the expression $\rho(x, t) = \rho(x - \int_0^t \frac{d}{dt} x_i(s) ds, 0)$. Similarly, if we choose a different reset policy, perhaps setting the rates to zero of those sources with higher rates, we change the congestion epoch duration slightly and the percentage of sources at a given rate no longer decays exponentially. In fact, if we adopt a "deterministic" reset policy where we reset the N/q sources with the highest rates, we will get q distinct rate populations of equal size and a constant epoch duration depending on q.

Finally, the mean-field approximation is just that, only an approximation of the true behavior of the mathematical model (itself an approximation). However, a more careful analysis of the behavior of N sources under the action of this random dynamical system, *not* assuming that N is large, reveals that the mean-field approximation is an excellent approximation of the system in Figure 11.

IV. CONCLUSION

In our work we set out to explore and quantify by how much RED reduces "global synchronization". Intuitively, it is a small step from understanding how sources may move in lock-step to conclude that a randomized drop policy will break the synchronization and hence eliminate aggregate periodic behavior.

Our first conclusion is that the intuition is wrong — while RED may prevent sources moving in lock-step, this is not enough to prevent the traffic rate on a link from oscillating, and hence wasting throughput for all flows.

Our second conclusion — aided by a simple analytical model — is that there is evidence that when buffers are small, a feedback-based congestion control algorithm has a strong propensity to cause aggregate periodic behavior, regardless of the queue management policy. In other words, randomized drop policies are unlikely to reduce aggregate periodic behavior. Even when the system is perturbed in a variety of ways, such as placing sources as very different distances from the bottleneck, traffic on the bottleneck link can still oscillate. Particularly with small buffers, flows in apparently very different states interact — and appear to conspire — to create aggregate periodicity. However, our results are based on a simple topology and a toy analytical model — it is premature to make any strong generalizations from our results. Further work is needed to determine how broadly these results apply.

Fortunately, although traffic on the link prior to the bottleneck router may continue to oscillate, with large enough buffers (equal to at least the bandwidth-RTT product), neither Drop Tail and RED lead to significant periodic behavior on the bottleneck link itself.

REFERENCES

[1] S. Bajaj et. al., "Virtual InterNetwork Testbed: Status and Research Agenda," USC/ISI Technical Report 98-678, July 1998.

[2] S. McCanne and S. Floyd, "UCB/LBNL/VINT Network Simulator - ns (version 2)," http://www-mash.cs.berkeley.edu/ns/

[3] K. Fall, S. Floyd, "Simulation-based comparisons of TCP Tahoe, Reno, and SACK TCP," *Computer Communication Review*, Vo. 26, No. 3, pp.5-21, 1996

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

[5] M. Allman and V. Paxson and W. Stevens, "TCP Congestion Control," Request for Comments 2581, April 1999

[6] B. Braden et. al., "Recommendations on Queue Management and Congestion Avoidance in the Internet," Request for Comments 2309, April 1998

[7] S. Shenker, L. Zhang, D. Clark, "Some Observations on the Dynamics of a Congestion Control Algorithm," *ACM Computer Communication Review*, Vol. 20, No. 4, pp.30-39, October 1990.

[8] L. Zhang, S. Shenker and D. Clark, "Observations on the Dynamics of a Congestion Control Algorithm: The Effect of Two-Way Traffic," *Proc. of ACM/SIGCOMM'91*, pp. 133-149, 1991.

[9] Y. Joo, V. Ribeiro, A. Feldmann, A. Gilbert, W. Willinger, "On the impact of variability on the buffer dynamics in IP networks," Proc. of 37th Annual Allerton conference on Communication, Control, and Computing, Sept. 1999

[10] A. Veres, M. Boda, "The Chaotic Nature of TCP Congestion Control," *Proc. of Infocom 2000*, Tel Aviv, Israel, March 2000.

APPENDIX A: Analysis of Toy Model

In this section we sketch the proofs of the results discussed in Section III.

1) Invariant configurations

Theorem 1:There is a unique density ρ_* that describes the state of N sources (for N large) immediately after resetting the rates of $r \cdot N$ sources (chosen uniformly at random) to zero such that after the next reset, reached at time $\Delta t = (r \cdot C)/N$ after the first reset, ρ_* is restored. This density is given by

$$\rho_*(x) = \begin{cases} r \cdot \sum_{l=0}^{\infty} (1-r)^l \cdot \delta_0 \cdot (x - l \cdot \Delta t), x \geq 0 \\ 0, x < 0 \end{cases}$$

In addition, the aggregate rate $A(t)$ is a periodic function with period $\Delta t = (r \cdot C)/N$, oscillating between a maximum value of C and $(1-r) \cdot C$.

Proof: The aggregate rate at time t is $A(t) = N \cdot \int (x \cdot \rho(x,t))\, dx$ where the density $\rho(x,t)$ describes the state the system is in at time t and the capacity constraint gives us the time t_n^- at which the rate reaches capacity C

$$C = A\left(t_n^-\right) = N \cdot \int (x \cdot \rho(x, t_n))\, dx$$

We obtain an iterated map on the density $\rho(x,t)$ and aggregate rate $A(t)$. Assume that t_n^+ is the time immediately after the n th reset, then the iterated map is

$$\rho\left(x, t_n^+\right) = r \cdot \delta_0(x) + (1-r) \cdot \rho(x - \Delta t_{n+1}, t_{n+1}^-)$$

$$A(t_{n+1}^-) = C = A(t_n^+) + N \cdot \Delta t_{n+1}$$

$$A(t_n^+) = A(t_{n+1}^+) = (1-r) \cdot C$$

where $\Delta t_{n+1} = t_{n+1} - t_n$. The last two conditions on the aggregate rate tell us that if a density $\rho_*(x)$ exists, then Δt satisfies $(1-r) \cdot C + N \cdot \Delta t = C$ or that $\Delta t = (r \cdot C)/N$.

The global attractor for the iterated map on $\rho(x,t)$ must therefore satisfy

$$\rho_*(x) = \begin{cases} r \cdot \delta_0(x) + (1-r) \cdot \rho_*(x - \Delta t), x \geq 0 \\ 0, x < 0 \end{cases} \quad (1)$$

Observe that we can translate condition specified in (1) into a condition on $\bar{\rho}_*$, the Laplace transform of ρ_*, and then we can expand in a uniformly convergent power series

$$\bar{\rho}_*(s) = \frac{r}{1 - (1-r) \cdot e^{-(s \cdot \Delta t)}} = r \cdot \sum_{l=0}^{\infty} (1-r)^l \cdot e^{-(s \cdot l \cdot \Delta t)}$$

Next we compute the inverse Laplace transform of $\bar{\rho}_*(s)$ by inverting each term in the power series term-by-term, giving us

$$\rho_*(x) = r \cdot \sum_{l=0}^{\infty} (1-r)^l \cdot \delta_0 \cdot (x - l \cdot \Delta t)$$

when $x \geq 0$ and $\rho_*(x) = 0$ for $x < 0$.

Note that if we take an iterative approach, we find the following theorem:

Theorem 2:If a fraction r of a system of N sources whose states are initially described by the density $\rho(x,0)$ follows the dynamics outlined above, then the density $\rho(x,t)$ converges weakly to the global attractor $\rho_*(x)$, independent of the initial conditions of the system. The density at time r_n after the n-th reset is

$$\rho(x, t_n) = r \cdot \sum_{l=0}^{\infty} (1-r)^l \cdot \delta_0 \cdot (x - l \cdot \Delta t) + (1-r)^n \cdot \rho(x - t_n, 0)$$

and the congestion epoch length is $\Delta t = (r \cdot C)/N$, after the first congestion epoch (whose length depends on the initial state of the system).

The careful reader might argue that allowing even an infinitesimal percentage of the sources' rates to grow arbitrarily large is not a physically relevant result. Let us now assume that the sources' rates cannot grow larger than a maximum rate D. That is, once a source increases its transmission rate to D, its rate remains fixed there until it experiences a loss. We have to assume that $C/N < D < C$ to avoid trivial and physically meaningless situations. We express this condition upon the source rate distribution mathematically by placing a point mass at rate $x = D$ whose weight is the total percentage of source rates that at time t are larger than $x = D$ and leaving unchanged the distribution for smaller rates. Let L be the largest integer such that $L \cdot \Delta t \leq D < (L+1) \cdot \Delta t$ for $\Delta t = (r \cdot C)/N$. Then we may conclude using arguments similar to the above theorems that there is an invariant distribution $\tilde{\rho}_*$ for D-bounded sources.

Corollary 1:The invariant distribution for D-bounded sources has the form

$$\tilde{\rho}_*(x) = r \cdot \sum_{l=0}^{L} (1-r)^l \cdot \delta_0 \cdot (x - l \cdot \Delta t) + m(r,L) \cdot \delta_0(x - D) \quad (2)$$

where $m(r,L) = r \cdot \sum_{l=L+1}^{\infty} (1-r)^l = (1-r)^{L+1}$ is the percentage of sources at rate D.

2) Stability of invariant configurations

We work with the more physically meaningful rate-bounded model as in Corollary 1 and assume $\varepsilon/N < D - L \cdot \Delta t$ for ease of calculation (if not, then the summation in (2) ends at $l = L+1$).

Lemma 1:If we jitter the duration of the i th congestion epoch by ε_i/N for ε_i a uniformly distributed value in

$[-\varepsilon,\varepsilon]$, *then only the duration of the* $(i+1)$ *st epoch is affected and* $\Delta t_{i+1} = \Delta t - (1-r)\cdot\varepsilon_i/N$, *where* $\Delta t = (r\cdot C)/N$.

Proof: Because we reset the source rates after an epoch duration $\Delta t_i = \Delta t + \varepsilon_i/N$, the total rate immediately before reset is allowed to grow to $A(t_i^-) = C + \varepsilon_i$ and immediately following reset is $A(t^+_i) = (1-r)\cdot(C+\varepsilon_i)$. We let the system of sources evolve as before until their total rate reaches capacity C. The duration of the $i+1$ st epoch is the time interval given by $C = A\left(t^+_i\right) + N\cdot\Delta t_{i+1} = (1-r)\cdot(C+\varepsilon_i) + N\cdot\Delta t_{i+1}$ or $\Delta t_{i+1} = (rC - ((1-r)\cdot\varepsilon_i))/N$. At this time (with no jitter), we reset a fraction r of the source rates and proceed as usual. Because at the $i+1$ st epoch, the total rate is C and we use the same evolution procedure and reset policy at the $i+2$ nd epoch, the duration $\Delta t_{i+2} = (rC)/N$ remains unaffected by the perturbation at the i th epoch.

Note that if we jitter the $i+1$ st epoch duration by ε_{i+1}/N, then the total rate before reset is $A(t^-_{i+1}) = C + \varepsilon_{i+1}$ (and this rate is not affected by the jitter at the i th epoch). We may then apply the lemma to the $(i+1)$ st epoch. Applying this lemma inductively, we may conclude:

Theorem 3:If we jitter each congestion epoch duration independently and let the system evolve, these perturbations do not propagate and force the system away from the invariant configuration with respect to the duration of the epochs. That is, the duration of each epoch varies only by ε/N, *the amount by which we perturb the duration, and on average* $\Delta t_i = \Delta t = (rC)/N$.

These calculations do not tell us if the rates of the perturbed system remain close to those of the original system however. We return to the setting of Lemma 1 and assume that the system has evolved long enough in time to have the form immediately after a reset

$$\rho(x,t) = r\cdot\sum_{l=0}^{L}(1-r)^l\cdot\delta_0\cdot(x - l\cdot\Delta t) + \tilde{m}_n(r,L)\cdot\delta_0(x-D)$$

where the mass $\tilde{m}(r,L)$ at the rate bound $x = D$ is

$$\tilde{m}(r,L) = r\cdot\sum_{l=L+1}^{\infty}(1-r)^l + (1-r)^{L+1}\cdot\int_0^{\infty}\rho(x,0)dx$$

Lemma 2:If we jitter the duration of the i *th congestion epoch by* ε_i/N *for* ε_i *a uniformly distributed value in* $[-\varepsilon,\varepsilon]$, *then the source rates change by* $max(((1-r)\cdot\varepsilon_i)/N, (r\varepsilon_i)/N)$.

Proof: The reader is asked to verify that if $\Delta t_i = \Delta t + \varepsilon_i/N$ and $\Delta t_{i+1} = \Delta t - ((1-r)\varepsilon_i)/N$, then before the $i+1$ st reset, the first rate population is at rate $\Delta t - ((1-r)\varepsilon_i)/N$ while all the others are at rate $k\Delta t + \varepsilon_i/N - ((1-r)\varepsilon_i)/N = k\Delta t + (r\varepsilon_i)/N$. In addition, for $k < L-1$ iterations after the perturbation at the i th epoch, the source rates have wandered from their original trajectory by at most $max(((1-r)\cdot\varepsilon_i)/N, (r\varepsilon_i)/N)$. After $L-1$ iterations proceeding the perturbation, the system returns to its initial trajectory.

By applying this lemma inductively, we may conclude:

Theorem 4:If we perturb each congestion epoch duration independently and let the system evolve, then the maximum deviation of the source rates from integer multiples of $\Delta t = (rC)/N$ *is* $r/N\cdot\sum_{k=0}^{L}\varepsilon_k$, *for* ε_k *i.i.d. uniform random variables on* $[-\varepsilon,\varepsilon]$.

We observe that if the maximum source rate D is on the order of C, then L is approximately N/r. Hence the maximum deviation is roughly $rL\varepsilon/N = \varepsilon$ and on average this deviation is zero since ε is a uniformly distributed random variable with mean zero.

APPENDIX B: RED parameters used in simulations

We use the same notation for the parameters as [4].

- Queue occupancy is measured in number of packets.
- Dropping probability is NOT increased slowly when average queue length exceeds maxthresh.
- When the algorithm decides to drop a packet, it drops from the tail.
- The algorithm doesn't favor any packet based on its type(control or data) or size.
- $q_w = 0.002$ (queue weight given to current queue size sample).
- $p_b = 0.1$ (Maximum probability of dropping a packet).