Comparative study of RED, ECN and TCP Rate Control

Prasad Bagal, Shivkumar Kalyanaraman, Bob Packer [1]
Department of ECSE, Rensselaer Polytechnic Institute,
Troy NY 12180-3590, U.S.A.
Ph: 518-276-8979; FAX: 518-276-2433
Corresponding author email: shivkuma@ecse.rpi.edu

INTENDED SYMPOSIUM: GI99 (Global Internet'99 Symposium)

Abstract

TCP congestion control is designed for network stability, robustness and opportunistic use of network buffer and bandwidth resources on an end-to-end per-connection basis. However its loss-based control strategy results in poor fairness properties which is the primary cause of the "World Wide Wait" problem. In this paper, we evaluate three proposed solutions for these problems - an improved drop scheme (RED), a bit-based explicit congestion notification scheme (ECN) and a scheme which explicitly and transparently controls TCP rate (Packeteer TCP rate control). Our studies indicate marked improvements in fairness as we move from RED through ECN to TCP rate control.

1 Introduction

TCP congestion control is designed for network stability, robustness and opportunistic use of network buffer and bandwidth resources on an end-to-end per-connection basis [9]. Using a robust technique to detect packet loss (timeout or triple-duplicate acks [18]), TCP infers congestion and trades off per-user goodput for network stability. Specifically, TCP throughput is known to be a function which is inversely proportional to the round trip time, the timeout delays and the square root of loss probability [15].

Besides stability and robustness, one could apply the following generic evaluation criteria to evaluate congestion control schemes. First, network operators evaluate performance by looking at a balance between high utilization of bottleneck links, low average queue length and low packet drop rate. Second, users of the network evaluate performance (assuming infinite flows) based upon per-flow goodput as seen by the application (approximated by the combination of average and standard deviation of goodput). In particular, users desire average goodput to be as high as possible, and the standard deviation in goodput close to zero. Based on these measures and the TCP throughput function [15], we observe that variance in parameters such as round trip times (RTT), drop probabilities or timeout delays leads to variance in per-flow goodput. Since packet drop characteristics affect all three parameters (queueing delay component of RTT, drop probabilities and timeout occurrences), the problem of TCP performance optimization is greatly affected by drop characteristics.

For example, we know that when a source experiences packet loss, it is an indication of a congestion event. But *the occurrence of congestion event at a bottleneck does not imply that at least one packet*

[1] Bob Packer is with Packeteer, Inc.

from all active flows are dropped, or should be dropped. This ambiguity introduces unfairness because some flows may back off while others do not. Further, *multiple packet loss events experienced by a source do not imply multiple congestion events in the path* - packet losses may be correlated [2]. Common TCP Reno-derived implementations further complicate this optimization problem because they do not filter loss-indications well enough to determine (a smaller set of) congestion events. A burst drop of three or more packets with nearby sequence numbers would result in multiple window decreases followed by a timeout. Given these types of issues in optimization of per-flow TCP goodput (and its variance across flows), it is worthwhile to consider a non-packet-drop based strategy as a primary method for TCP performance optimization.

The Internet Engineering Task Force (IETF) and independent researchers have proposed several improvements to TCP/IP-based control at the transport and network layers. The Random Early Detection (RED) buffer management strategy was designed to break the TCP synchronization problem, primarily through randomization and non-bursty early packet dropping (before the buffer was full). An example of transport layer enhancement is the selective acknowledgments (SACK) scheme [13]. The Explicit Congestion Notification (ECN, one bit explicit feedback) scheme [16] requires both end-to-end and network support. Network-based enhancements are aimed at improving fairness and throughput. These include mechanisms like scheduling [5], improved packet discard policies [7, 6, 12, 11] and explicit rate control of TCP [17, 14]. The end-to-end proposals aim to provide better filtering to detect congestion events from loss events, improved retransmission, and attempt to reduce the occurrence of timeouts. Providing fairness and queueing delay control have required some form of network-based support.

In this paper we shall study the fairness, queueing delay and loss control characteristics of three proposed schemes: RED, ECN and Packeteer's TCP rate control (a network-based solution which controls the left and right edges of the TCP window, and shapes the TCP acknowledgment stream [14]). Our studies indicate marked improvements in fairness (as measured by the standard deviation in goodput and covariance) without significant tradeoffs in other metrics as we move from RED through ECN to TCP rate control. All schemes control bottleneck queueing delay, but trading off other measures such as drop rate, utilization and fairness, with TCP rate control showing the best performance. However for short transfers on low speed links, RED and ECN tradeoff fairness to achieve improvement in the total number of transfers as measured over a medium sized simulation time-window. TCP rate control and RED offer the fastest route to immediate and widespread deployment. The minimal state requirements and protocol transparency of RED allow it a greater available space of deployment scenarios.

This paper is organized as follows. Section 2 gives a description of the TCP rate control strategy (RED and ECN are described in [7] and [16]). The performance analysis section (Section 3) is split into sections describing metrics, parameters and configurations (Sections 3.1, 3.2) and simulation results (Section 4). We summarize our observations and conclude in Section 5. An extended version of the paper is available as [19].

2 TCP Rate Control

TCP rate control is a technique where the rate of a TCP flow is directly and explicitly controlled [17, 14]. Packeteer's rate control solution (product called "Packetshaper^{TM}") exploits the fact that **TCP's rate is determined by a) the rate of acknowledgments, b) the acknowledgment number and, c) the receiver maximum window size fields** (the last two variables determine the left and right edges of the TCP window).

Specifically, *given a constant window size, the TCP rate is equal to the rate of the "ack clock"* i.e. the stream of acknowledgments. Control of the ack rate also smoothes out burstiness in TCP transmission. Also, the packetshaper can tradeoff acknowledgment queues in the reverse direction (which can be optimized even further by compressing the information) for packet queues in the forward direction [17].

The discussion above assumed a constant TCP window. But, the TCP window is not a constant - *the right edge of the window is controlled by the CWND (congestion window) variable and the receiver window field in TCP acks; and the left edge of the window is controlled by the acknowledgment number field in TCP acks.* The packetshaper therefore uses the two fields - the receiver window field and acknowledgment number field - to control the size of the window. The control of the window in addition to ack shaping allows packet shaper to control aggregate queueing delays as well as individual TCP throughput at a fine granularity.

In addition, when placed at key bottlenecks in the system, the packetshaper can *dynamically divide the available capacity among the contending flows in a fair manner using a combination of policy rules, a scheduling algorithm and a dynamic rate calculation algorithm.* Specifically, allocated capacity temporarily unused by a bottlenecked flow is fairly distributed to other contending flows to achieve max-min fairness [4, 10, 17]. Observe that using this technique, packetshaper can rapidly and accurately adapt to dynamic capacity changes. *The calculated rate allocation is enforced by a) controlling the ack rate, and b) control of window edges after translation of rate into a window value.* Rate-to-window translation requires knowledge [17] of round trip times which can be estimated at the packetshaper.

The remarkable feature is that TCP rate can be completely controlled only by training the ack stream, and the solution is transparent to end systems (TCP) or routers. Packetshaper works best when the TCP packet and ack flow are accessible to it, and can perform the core functions even under limited asymmetry (eg: ack flow alone accessible). Since it requires reading and writing into TCP headers, these headers should be accessible (eg: not encrypted or authenticated). Though it maintains per-flow state, the storage and state requirements scale very well (products supporting over 20,000 simultaneous flows are now in operation). The solution is applicable to network edges rather than cores. In this deployment space, it is as easily deployable as RED and is transparent to hosts and routers. Note that ECN does not share the advantage of deployability since it requires changes to host TCP implementations.

3 Performance Analysis

For performance analysis purposes, we consider the system (of TCP flows in this case) as a black box to which is input a set of ***parameters*** and workloads (parameters include the choice of the scheme, configurations etc) and the output is a set of ***metrics*** which evaluate the tradeoff among various resource constraints in the system.

3.1 Metrics

As mentioned briefly in the introduction section, we classify metrics into two major categories: user metrics and operator metrics.

Operator Metrics: Average link Utilization : low link utilization, given adequate load is unacceptable.

Average Queue Length : Low average queue lengths imply lower average queueing delay experienced by participating connections. Prefer low queue lengths combined with high utilization.

Maximum Queue Length : Very high maximum queue lengths indicate high buffer requirements.

Packet drop rate : Packets dropped represent wasted bandwidth and buffer resources on upstream links. Packet drop also has a high cost for users, but is unavoidable during congestion.

User Metrics for Infinite (long) Transfers: The user is interested in his/her per-flow goodput (which excludes retransmissions) for infinite flows which we approximate by:

Average (per-flow) Goodput : (μ goodput) This quantity should be as high as possible. Ideally, for a single bottleneck, average goodput times number of sources should equal the product of utilization and bandwidth, where utilization is as close to 100% as possible.

Standard deviation in (per-flow) goodput: (σ goodput) This quantity is a rough measure of fairness. Ideally, for a single bottleneck with infinite transfers, this metric should be close to zero.

Covariance: In our simulations, we also consider the covariance which ratio of standard deviation to average (goodput) and prefer a smaller ratio. The covariance is only used for convenience.

User Metrics for Short Transfers: The user is interested in per-transfer response time. As before, we use the following proxies for this quantity.

Average time for a transfer (across all transfers): Indicative per-transfer response time (along with the standard deviation)

Standard deviation in time for transfer (across all transfers): Indicative of per-transfer response time (along with with the mean).

Our simulation results are presented in a tabular format with user and operator metrics separated and significant results highlighted.

3.2 Parameters (factors) and configurations

Figure 1 gives the basic configuration template that we used in our simulations. It contains a single bottleneck shared by a set of unidirectional TCP flows and allows examination of multiple dimensions. We present only a representative set of results here; more results are documented in [19].

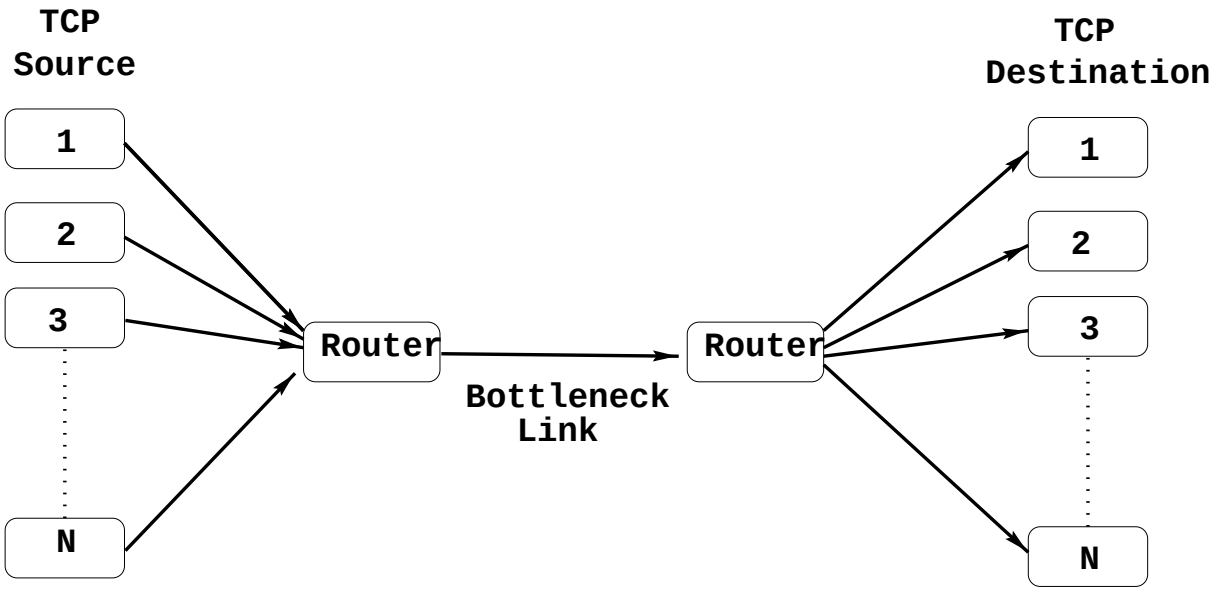

Figure 1: Configuration template used in Simulation

4 Simulation Results

This section reports simulation results using homogeneous RTT configuration, heterogeneous RTT configurations and with a workload of short transfers. These dimensions are sufficient to clearly differentiate between the performance of the three schemes. Additional parameter dimensions (staggered flow start times, shorter RTTs and different number of flows) are explored in [19], which also report significant performance issues.

4.1 Dimension: RED vs ECN vs TCP rate control

This section presents a preliminary look at the parameter dimension of schemes evaluated (RED vs ECN vs TCP rate control) while keeping most of other parameter dimensions (except link speeds) simple and constant. Specifically, we use 100 connections, one-way distance of 3000 km (corresponding to propagation delays of 15 ms one way), homogeneous RTT configuration. The

link speeds vary from 28 Kbps to 150 Mbps. Our primary observation in this section is that *good performance in terms of provider metrics does not imply good performance in terms of user metrics* (esp. in drop-based schemes such as RED). Due to the symmetric nature of the configuration, problems in ECN do not surface.

4.1.1 RED (homogeneous RTTs)

In table 1 we observe the operator and user metrics when the bottleneck implements Random Early Detection. This simulation was conducted with 100 connections, one-way distance of 3000 km (corresponding to propagation delays of 15 ms one way), homogeneous RTT configuration. The link speeds vary from 28 Kbps to 150 Mbps.

	Operator Metrics			*User Metrics*		
Speed Mbps	**Util** Percent	**μ Q/Max Q** Pkts/Pkts	**Drop** Pkts	**μ Goodpt** Mbps	**σ Goodpt** Mbps	**Covar**
0.028	93.75	30.75/61	439	0.00082	0.00000	0.000
0.056	92.67	31.59/63	405	0.00098	0.00049	0.500
0.128	98.12	30.19/62	534	0.00206	0.00132	0.641
0.256	99.18	28.37/62	553	0.00349	0.00307	0.880
0.384	99.41	29.55/61	638	0.00487	0.00279	0.573
1.5	99.66	23.82/63	1002	**0.01444**	**0.01558**	**1.078**
10	99.50	18.59/63	2215	0.08652	0.05919	0.684
45	98.51	13.29/63	3818	0.37506	0.11369	0.303
150	96.57	9.86/63	3637	1.14209	0.35048	0.307

Table 1: RED; 100 sources; Homogeneous RTT; One-way distance = 3000 Km.

Observe that RED satisfies operator metrics: high utilization and low queue lengths. A non-trivial number of packets were dropped in all cases which included burst drop instances. From the user perspective, the aggregate goodput (which is average goodput multiplied by 100, taken as a fraction of bottleneck link speed) degrades with increase in bottleneck link speed. Lower values of goodput indicate more retransmission overhead. The covariance and standard deviation in goodput is consistently high indicating unfairness. The highlighted entry shows that the highest covariance occurs for a 1.5 Mbps link. This is due to the effects of burst drop concentrated on a small subset of sources and its interaction with the TCP Reno implementation. The simulation results show that RED has difficulty spreading out packet drops among even a moderate number of active connections.

4.1.2 ECN (homogeneous RTTs)

Table 2 shows the performance of ECN under the same set of conditions as the previous simulation set.

With the introduction of ECN, we immediately remove the effects of packet drop and the large

	Operator Metrics			*User Metrics*		
Speed Mbps	**Util** Percent	μ **Q/Max Q** Pkts/Pkts	**Drop** Pkts	μ **Goodpt** Mbps	σ **Goodpt** Mbps	**Covar**
0.028	98.25	208.60/441	0	0.00131	0.00075	0.573
0.056	98.90	233.01/487	0	0.00187	0.00079	0.421
0.128	99.33	275.04/542	0	0.00269	0.00057	0.212
0.256	99.57	292.08/502	0	0.00341	0.00081	0.237
0.384	99.68	**327.63/557**	0	0.00441	0.00083	0.187
1.5	99.90	238.64/295	0	0.01678	0.00072	0.043
10	99.98	228.57/264	0	0.10167	0.00081	0.008
45	99.99	103.01/193	0	0.44987	0.00061	0.001
150	96.53	38.21/174	0	1.40727	0.02705	0.019

Table 2: ECN; 100 sources; Homogeneous RTT; One-way distance = 3000 Km

variance effects caused by it. The removal of packet drop effects are also partially due to the better filtering of congestion indications at the ECN-TCP source compared to TCP Reno. Except for long queueing delays experienced (which can be explained as a result of ECN marking on received packet at the tail of the queue rather than marking the packet in the front of the queue), all metrics show picture perfect performance. But as we shall see this performance is partly due to the homogeneous RTT configuration.

4.1.3 TCP Rate Control (homogeneous RTTs)

Table 3 shows the performance of TCP rate control under the same set of conditions as the previous two simulation sets. Our model of TCP rate control did not include several features targeted for lower speed links. The result of these modeling limitations shows up as high queueing delays for slower speed bottlenecks. But, for medium and higher speed cases the performance is excellent in terms of all metrics.

4.2 Dimension: Homogeneous vs Heterogeneous RTTs

This section looks at performance of the three schemes when the round trip times are heterogeneous. These simulations are conducted with 10 sources (not 100 sources as used in the last three sets). The flows were grouped into four sets and all flows from a given set had an one-way distances of 1002Km, 2000Km, 3000Km and 5000Km respectively. The link speeds vary from 28 Kbps to 150 Mbps.

This section clearly differentiates the benifits of explicit TCP rate control when compared to ECN and RED. Specifically, TCP rate control provides fairness (in terms of user metrics) even when RTTs are heterogeneous, without deteriorating any of the other metrics.

	Operator Metrics			*User Metrics*		
Speed Mbps	**Util** Percent	**μ Q/Max Q** Pkts/Pkts	**Drop** Pkts	**μ Goodpt** Mbps	**σ Goodpt** Mbps	**Covar**
0.028	98.25	207.20/437	0	0.00128	0.00074	0.576
0.056	98.90	230.77/481	0	0.00183	0.00080	0.434
0.128	99.33	272.49/538	0	0.00269	0.00057	0.212
0.256	99.57	283.83/488	0	0.00337	0.00081	0.241
0.384	99.68	320.35/540	0	0.00428	0.00059	0.138
1.5	99.90	184.92/200	0	0.01656	0.00080	0.048
10	99.98	161.71/199	0	0.10130	0.00077	0.008
45	99.99	33.80/193	0	**0.44981**	**0.00032**	**0.001**
150	99.84	41.59/193	0	1.48251	0.02731	0.018

Table 3: TCP Rate Control; 100 sources; Homogeneous RTT; One-way distance = 3000 Km

4.2.1 RED (heterogeneous RTTs)

Table 4 lists the performance metrics with RED in a heterogeneous RTT configuration.As explained in the introductory sections, TCP throughput is an inverse function of RTT as well as loss probability and timeout delays. The introduction of heterogeneous RTTs adds to the variance in per-flow goodput (and increases the co-variance) in TCP with or without packet loss. We expect the variance to be larger when both packet loss and RTT variance is present.

	Operator Metrics			*User Metrics*		
Speed Mbps	**Util** Percent	**μ Q/Max Q** Pkts/Pkts	**Drop** Pkts	**μ Goodpt** Mbps	**σ Goodpt** Mbps	**Covar**
0.028	97.22	30.08/46	11	0.00401	0.00113	0.281
0.056	98.55	28.24/46	52	0.00623	0.00111	0.179
0.128	99.36	19.81/45	68	0.01581	0.00615	0.389
0.256	99.68	19.29/45	89	0.02703	0.01463	0.541
0.384	99.16	18.30/45	114	0.03917	0.01327	0.339
1.5	99.63	16.97/44	242	0.14788	0.05646	0.382
10	99.10	13.96/42	743	0.91430	0.42644	0.466
45	93.33	10.00/47	951	**2.65288**	1.16895	0.441
150	**79.88**	7.79/50	1079	**5.68535**	2.83307	0.498

Table 4: RED; 10 sources; Heterogeneous RTTs

Interestingly, the utilization remains high in most cases (except the 150 Mbps case), but goodputs deteriorate sharply (aggregate goodput is less than 60% of bottleneck speed in the cases highlighted). Standard deviation in goodput and covariance remain high indicating unfairness in addition to lower average goodput. This further suggests that RED optimizes operator metrics, trading off user-metrics under these conditions.

4.2.2 ECN (heterogeneous RTTs)

Table 5 shows the performance of ECN under similar conditions as the last simulation set (heterogeneous RTTs, 10 sources).

	Operator Metrics			*User Metrics*		
Speed Mbps	**Util** Percent	μ **Q/Max Q** Pkts/Pkts	**Drop** Pkts	μ **Goodpt** Mbps	σ **Goodpt** Mbps	**Covar**
0.028	97.22	33.02/56	0	0.00401	0.00113	0.281
0.056	98.57	41.61/69	0	0.00614	0.00123	0.200
0.128	99.37	62.00/108	0	0.01515	0.00303	0.200
0.256	99.68	62.46/86	0	0.02408	0.00659	0.273
0.384	99.79	42.64/70	0	0.03965	0.00549	0.138
1.5	99.95	35.74/79	0	0.15262	0.02161	0.142
10	99.99	26.60/68	0	1.00246	0.37674	0.376
45	100.00	19.32/69	0	4.50278	4.18975	0.930
150	100.00	16.50/76	0	15.00642	**19.37648**	**1.291**

Table 5: ECN; 10 sources; Heterogeneous RTTs

These simulations show that while ECN is capable of satisfying all the operator metrics in this heterogeneous RTT case, it is not capable of reducing the variance in goodput (unfairness). The covariance in fact increases as the link speeds increase. This points to the insufficiency of single-bit feedback in achieving fairness goals. The negative effects of packet drop are removed however. We also note the marked decrease in queue lengths with the decrease in the number of active connections.

4.2.3 TCP rate control (heterogeneous RTTs)

Table 6 shows the corresponding performance of TCP rate control under these circumstances.

We observe the superiority of the TCP rate control compared to ECN and RED in this configuration. The TCP sources achieve high fairness and goodput almost independent of the RTTs. Performance in terms of all metrics is excellent. The reason TCP rate control avoids both the effects of heterogeneous RTTs and packet loss is because it uses explicit window control and hence does not conform to the Padhye formula for TCP throughput. This allows it to achieve fair allocations independent of RTT variance of participating flows. Though explicit control, packet drops are also avoided.

4.3 Dimension: Short transfers vs Long transfers

In this section we look at the performance of the schemes with respect to short file transfers. The same configuration template (Figure 1) is used in these simulations. Flows are grouped into four sets and the start time of every set is staggered by 250 ms. Every flow sends 10K bytes (10 packets)

	Operator Metrics			*User Metrics*		
Speed Mbps	**Util** Percent	μ **Q/Max Q** Pkts/Pkts	**Drop** Pkts	μ **Goodpt** Mbps	σ **Goodpt** Mbps	**Covar**
0.028	97.14	29.60/48	0	0.00328	0.00063	0.194
0.056	98.55	36.01/61	0	0.00500	0.00057	0.115
0.128	99.37	19.20/25	0	0.01286	0.00052	0.041
0.256	99.68	17.39/20	0	0.02711	0.00093	0.034
0.384	99.79	18.41/20	0	0.03981	0.00054	0.014
1.5	99.95	16.59/20	0	0.15106	0.01089	0.072
10	99.99	11.39/20	0	1.00115	0.44187	0.441
45	100.00	13.61/34	0	4.50401	1.01488	0.225
150	99.96	13.89/94	0	**15.00257**	**1.37127**	**0.091**

Table 6: TCP rate control; 10 sources; heterogeneous RTTs

and closes the connection and transfer. After a pause of 250 ms, the same source would "re-open" the connection (with parameters set to initial values) and send another 10K bytes (10 packets) and so on. Though Web transfers are typically short, this model is not an accurate model of the world wide web (WWW). A similar model has been used in the past [1].

Our primary observation in this configuration is that the tradeoff in fairness by RED and ECN in low speed bottleneck configurations leads to larger aggregate number of completed transfers compared to TCP rate control. At higher bottleneck speeds, there is no difference in performance.

4.3.1 RED (Short transfers)

	Operator Metrics			*User Metrics*		
Speed Mbps	**Util** Percent	μ **Q/Max Q** Pkts/Pkts	μ T. time Milli Secs	σ **T. time** Milli Secs	**Covar**	**# Trans**
0.256	99.29	28.90/51	9527	929.2244	0.098	9
0.384	99.49	28.93/51	8852	2127.2276	0.240	27
1.5	99.39	25.20/49	3763	2631.0244	0.699	227
10	97.87	13.07/50	588	**1077.7262**	1.832	**1457**
45	80.98	3.05/44	143	394.4417	2.750	2513

Table 7: RED; 100 sources; Repeated Short Transfers

Table 7 shows the results for simulations with RED. We observe that the average transfer time reduces with increase in bandwidth. But the effective bandwidth consumed by an average transfer is much less than available bandwidth. This is partially due to the fact that most of the transfer time is spent in slow start, and in timeouts. The standard deviation of transfer time (and covariance) is large indicating that some transfers were able complete at the expense of other transfer (unfairness).

4.3.2 ECN (Short transfers)

Table 8 shows the simulation results of short transfers with ECN.

	Operator Metrics			*User Metrics*		
Speed Mbps	**Util** Percent	μ **Q/Max Q** Pkts/Pkts	μ T. time Milli Secs	σ **T. time** Milli Secs	**Covar**	**# Trans**
0.256	98.80	312.53/528	9625	279.5135	0.02904	3
0.384	99.16	331.81/547	9430	1336.5891	0.14174	8
1.5	99.57	197.25/249	4172	1585.0223	0.37991	164
10	99.84	117.24/189	537	**109.2574**	0.20328	**1191**
45	83.71	4.28/51	123	6.4239	0.05243	2600

Table 8: ECN; 100 sources; Repeated Short Transfers

We note that the average response times in slower speed configurations are slightly worse than RED because, in spite of not having loss-triggered effects, the queueing delays add significantly to the response time. Otherwise ECN performance scaled well with increase in link speeds allowing a larger number of transfers and reduction in covariance and standard deviation of transfer time.

4.3.3 TCP rate control (Short transfers)

The results for TCP rate control in this situation are shown in table 9.

	Operator Metrics			*User Metrics*		
Speed Mbps	**Util** Percent	μ **Q/Max Q** Pkts/Pkts	μ T. time Milli Secs	σ **T. time** Milli Secs	**Covar**	**# Trans** Number
0.256	98.80	283.89/470	-	-	-	0
0.384	99.16	289.10/485	-	-	-	0
1.5	99.35	202.13/309	3845	1375.3581	0.35770	186
10	99.74	151.67/244	**538**	**108.9222**	**0.20251**	**1198**
45	83.41	3.82/67	125	16.5318	0.13245	2600

Table 9: TCP rate control; 100 sources; Repeated Short Flows

Interestingly, at slower speeds, the TCP rate controller results in a smaller number of transfers completed. This is because, the rates of all contending transfers are equally reduced (for fairness) and each of them take longer. Also the simulations were run for 10s - a limited observation window - which was not long enough for these transfers to complete.

On the other hand the ECN and RED schemes traded off fairness (which they could not achieve) for increase in number of transfers completed. Specifically the congestion response by a subset of transfers gives an opportunity for other transfers to grab the available bandwidth at line speed (without congestion back off). However at higher speeds the transfer times of rate control are as good as ECN.

5 Summary and Conclusions

TCP throughput is known to be a function which is inversely proportional to the round trip time, the timeout delays and the square root of loss probability [15]. Variance in any of these component factors introduces variance in TCP throughput (and goodput) resulting in unfairness. Variance in more than one factor has a multiplicative effect on variance in throughput.

Packet drops (esp. burst drops) have a multiplier effect especially in configurations with heterogeneous RTTs. RED and ECN cannot control this variance even though ECN does not incur packet drops (in the best case). The explicit TCP rate control which controls the window edges of TCP transparently and shapes the ack rate provides best possible fairness, to a great extent removing the dependency of throughput on factors such as drop probability, RTT and timeouts. We observed that fairness in low speed configurations can lead to a reduction in the number of transfers completed during a small observation window, even though links are optimally utilized. A larger set of results are reported in [19].

TCP rate control and RED offer the fastest route to immediate deployment. But the minimal state requirements and protocol transparency of RED allow it a greater available space of deployment scenarios (core as well as edge routers).

References

[1] Hari Balakrishnan, Venkata Padmanabhan, Srini Seshan, Mark Stemm and Randy H. Katz, "TCP Behavior of a Busy Internet Server: Analysis and Improvements," *Proceedings of IEEE Infocom*, San Francisco, CA, USA, March 1998. http://www.cs.berkeley.edu/ hari/papers/infocom98.ps.gz

[2] J. Bolot and A. Vega-Garcia, "Control Mechanisms for packet audio in the Internet," *Proceedings of IEEE Infocom'96,* 1996.

[3] B. Braden, D. Clark, J. Crowcroft, B. Davie, S. Deering, D. Estrin, S. Floyd, V. Jacobson, G. Minshall, C. Partridge, L. Peterson, K. Ramakrishnan, S. Shenker, J. Wroclawski, L. Zhang, "Recommendations on Queue Management and Congestion Avoidance in the Internet," *Internet RFC 2309*, April 1998.

[4] Anna Charny "An Algorithm for Rate Allocation in a Packet-Switching Network with feedback", Masters thesis. MIT 1994

[5] A. Demers, S. Keshav, and S. Shenker, "Analysis and Simulation of a Fair Queueing Algorithm," *Internetworking: Research and Experience*, Vol. 1, 1990, pp. 3-26.

[6] W. Feng, D. Kandlur, D. Saha, K. Shin, "Techniques for Eliminating Packet Loss in Congested TCP/IP Networks," *U. Michigan CSE-TR-349-97,* November 1997.

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

[8] Sally Floyd, "TCP and Explicit Congestion Notification", *ACM Computer Communication Review*, Vol. 24, No. 5, October 1994, pp. 10-23. ftp://ftp.ee.lbl.gov/papers/tcp_ecn.4.ps.Z

[9] V. Jacobson, "Congestion Avoidance and Control," *Proceedings of the SIGCOMM'88 Symposium*, pp. 314-32, August 1988.

[10] S. Kalyanaraman, "Traffic Management for the Available Bit Rate (ABR) Service in Asynchronous Transfer Mode (ATM) networks" *Ph.D. Dissertation*, Dept. of Computer and Information Sciences, The Ohio State University, August 1997.

[11] Vijay Kumar, T.V. Lakshman, D. Stiliadis, "Beyond Best Effort: Router Architectures for the differentiated services of tomorrow's Internet," *IEEE Communications Magazine*, Vol. 36, No. 5, May 1998, pp. 152-164.

[12] Dong Lin and Robert Morris, "Dynamics of Random Early Detection," *Proceedings of SIGCOMM'97*, August 1997.

[13] M. Mathis, J. Mahdavi, S. Floyd, A. Romanow, "TCP Selective Acknowledgement Options," *Internet RFC 2018*, October 1996.

[14] Packeteer Inc., White papers on TCP rate control, http://www.packeteer.com/tcprate/

[15] Jitendra Padhye, Victor Firoiu, Don Towsley, and Jim Kurose, "Modeling TCP Throughput: A Simple Model and its Empirical Validation," *Proceedings of SIGCOMM'98*, Vancouver, August 1998.

[16] K.K. Ramakrishnan, S. Floyd, "A proposal to add Explicit Congestion Notification (ECN) to IPv6 and to TCP," *IETF Internet Draft*, November 1997, Available as http://ds.internic.net/internet-drafts/draft-kksjf-ecn-00.txt

[17] Ramakrishna Satyavolu, Ketan Duvedi, Shivkumar Kalyanaraman, "Explicit rate control of TCP applications," *Submitted at IWQoS'99,* 1999. http://www.ecse.rpi.edu/Homepages/shivkuma/research/papers/iwqos99-rate.ps

[18] W. Richard Stevens, "TCP/IP Illustrated, V. 1," *Addison-Wesley, Reading, MA*, 1995.

[19] P. Bagal, S. Kalyanaraman, B. Packer, "Comparative study of RED, ECN and TCP Rate Control," Technical Report, Dept of ECSE, RPI, 1999. http://www.ecse.rpi.edu/Homepages/shivkuma/research/papers/final.ps