

On the Resource Efficiency of Explicit Congestion Notification

Kostas Pentikousis and Hussein Badr

Department of Computer Science, Stony Brook University,
Stony Brook, NY 11794-4400, USA
{Kostas, Badr}@CS.StonyBrook.Edu

Abstract. Explicit Congestion Notification (ECN) for IP networks has received considerable attention in recent years, and has been shown to improve TCP goodput. Previous studies have centered on scenarios in which TCP with ECN (TCP/ECN) traffic competes with ECN-unaware traffic. This paper presents case studies in which moderately short flows are all ECN-capable, and compare them with the corresponding cases where the flows are ECN-unaware, running over drop tail and Random Early Detection routers. Using transmission overhead metrics, we show that TCP/ECN uses network resources more efficiently. We also consider the case of battery-operated devices and show that TCP/ECN is more power conserving than standard TCP. An unexpected outcome of our experiments is that goodput does not improve in an all-TCP/ECN environment.

1 Introduction

In the Transmission Control Protocol (TCP), senders use packet drops as a decisive indication of congestion in the network. Upon detection of segment loss, the TCP sender slows down its sending rate in an attempt to prevent congestion collapse [1]. Dropping packets to indicate congestion is a wasteful use of network resources. Furthermore, by equating segment loss with network congestion, TCP fails to perform well in networks where random errors are introduced by faulty hardware, or because the transmission media are less reliable than copper cables or fiber. Consequently, Explicit Congestion Notification (ECN) for IP networks [12] was proposed in order to provide TCP senders with clear indication of prevailing congestion in the network. ECN has received a considerable amount of attention in recent years, because it can help to increase network utilization and improve TCP performance (under traditional and emerging network technologies) [2][6][13][14], and because it might enable the development of quality of service differentiation schemes in IP networks [8].

This paper focuses on the resource efficiency of ECN in IP networks. Although one may intuitively surmise that there should be gains with ECN since packet drops are largely avoided, this has not been formally investigated, let alone quantified. We show that in an IP network where all hosts support ECN, network resources are used more efficiently with respect to criteria not considered in previous studies, which

tend to focus primarily on throughput/goodput; for example, transmission overhead from dropped and duplicated packets may be reduced by as much as 70%. Conversely, a significant, unexpected conclusion from our work is that average TCP throughput/goodput is not improved with respect to relatively short flows.

2 Related Work

ECN is integrally dependent on an active queue management (AQM) mechanism [4], which determines when packets should be marked [12]. In practice, in most studies published to date, the AQM underlying ECN has been Random Early Detection (RED) [7]. Slim and Ahmed [13], for example, use a Linux-based test bed network to study the performance advantage of TCP with ECN (TCP/ECN) in terms of throughput for both bulk and transactional transfers. Key findings include: increased relative advantage for TCP/ECN over standard TCP when congestion levels increase; limited number of retransmission when ECN is employed; and smaller amount of time spent in error recovery.

Zhang and Qiu [14] evaluate TCP performance under RED and drop tail (DT), examining a variety of scenarios, including long and short transfers, different round trip times (RTTs), a mix of TCP and UDP traffic, and two-way traffic. They also use TCP goodput as their metric of ECN efficiency. Though focusing mainly on RED with packet drops, they find that when n TCP/ECN senders compete with n ECN-unaware TCP senders, the former achieve up to 30% better goodput than the latter.

More recently, Athuraliya *et al.* [2] present an evaluation of TCP/ECN as part of a study of their proposed Random Exponential Marking (REM) AQM. They focus mainly on illustrating that REM performs better than RED, and that ECN can help to further reduce losses in the network while preserving high goodput. They consider the case where infrequent random drops are happening in the network (thereby roughly simulating errors in a wireless environment), and conclude that ECN can help to increase TCP's performance in hybrid wired/wireless networks [11].

A common, determinant characteristic of the published literature known to us is that it examines scenarios in which TCP/ECN flows compete with ECN-unaware ones. The literature makes clear that, under these circumstances, the TCP/ECN sender is virtually assured better goodput. All other things being equal, a TCP/ECN sender possesses more information about network conditions and avoids decreasing its congestion window more than once per window of data [12].

Another point worth highlighting is that most studies use “background”, non-ECN traffic in order to maintain a level of congestion that ensures routers operate in the “RED region”: that is, with average queue sizes always within the range of values for which random early marking is in effect. This permits ECN to display its best potential with respect to the ECN-capable flows present. However, it is known that tuning RED parameters so that routers in real networks, experiencing a variety of traffic mixes, are always operating in the RED region is difficult [5][9].

This paper presents case studies using simulation in which there is no background traffic. Traffic in our studies is ECN-capable, and is not constructed to ensure any

particular, *a priori* relationship with respect to the RED parameters in effect. We also focus on moderately short (100 KB) transfers, typical of, for example, web browsing. To the extent that a file transfer may terminate before the full potential of the feedback marking mechanism takes effect, shorter downloads are less favorable to ECN than longer ones. Our aim is to evaluate TCP/ECN performance under circumstances that are not necessarily inclined in its favor. The ECN cases are then compared to the corresponding cases where the traffic is ECN-unaware and running under, on the one hand, DT and, on the other, RED.

3 Experimental Configuration

Our study of TCP/ECN performance was conducted with the widely used ns-2 simulator [10]. Fig. 1 depicts the network topology, representing a situation in which multiple clients access a web server. Our network topology is similar to that used in many protocol evaluations, including the ones discussed above.

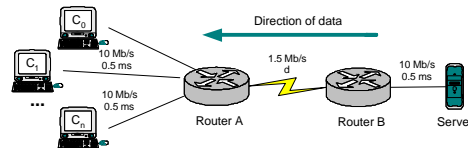


Fig. 1. Simulation topology

A fixed number of clients (receivers) n are each connected to router A through a 10 Mb/s link with a propagation delay of 0.5 ms. A server is connected to router B, also with a 10 Mb/s link and 0.5 ms propagation delay. Router B is connected to router A by a 1.5 Mb/s link (the “bottleneck link”). We experimented with a variety of propagation delays d for this link, ranging from 2 to 20 ms.

The server employs TCP Reno with a maximum segment size (MSS) of 1460 bytes. The clients use the delayed acknowledgements algorithm and advertise an initial receiver window of 64 KB [1]. Data transfers are unidirectional. Starting at time 0, each client initiates a 100 KB file transfer. When a client receives its file, it immediately initiates a new transfer for another 100 KB file. The three-way TCP connection-establishment handshake is simulated for every file transfer. An individual experiment terminates when the n parallel sets, each of 11 serial transfers, successfully complete.

We experiment with different values for the size of the router B output queue at the bottleneck link (*i.e.* in the direction of the data flow), using DT, and RED with and without ECN support. All other buffer queues in the network are DT, but in fact are adequately provisioned to ensure that no drops occur. This is in order to isolate the impact on performance induced by the router B queue management mechanism.

3.1 Simulation Methodology

As already mentioned, an individual experiment consists of n parallel sets, each composed of 11 serial file transfers, for a total of $11n$ transfers. Although the n sets start simultaneously, “synchronization” between them is quickly lost: first, because the SYN’s of the initial n connection establishment phases are serialized as they pass through the routers on their way to the server; and secondly because the $11n$ transfers undergo different experiences at the bottleneck router, yielding a rich mixture and variety of TCP congestion and error-recovery responses. This, for example, is illustrated in Fig. 2: the distribution of ECN marks across the possible sequence numbers is close to uniform for the vast majority of experiments. At any given instant of an experiment (at least until the first of the n parallel sets completes its 11 transfers) there are n simultaneous, ongoing downloads at different phases of progress and TCP induced dynamics. As such, the $11n$ transfers provide a rich sampling of possible outcomes. The results reported here, mostly averages for a single 100-KB file download are based on aggregation over this sample of size $11n$.

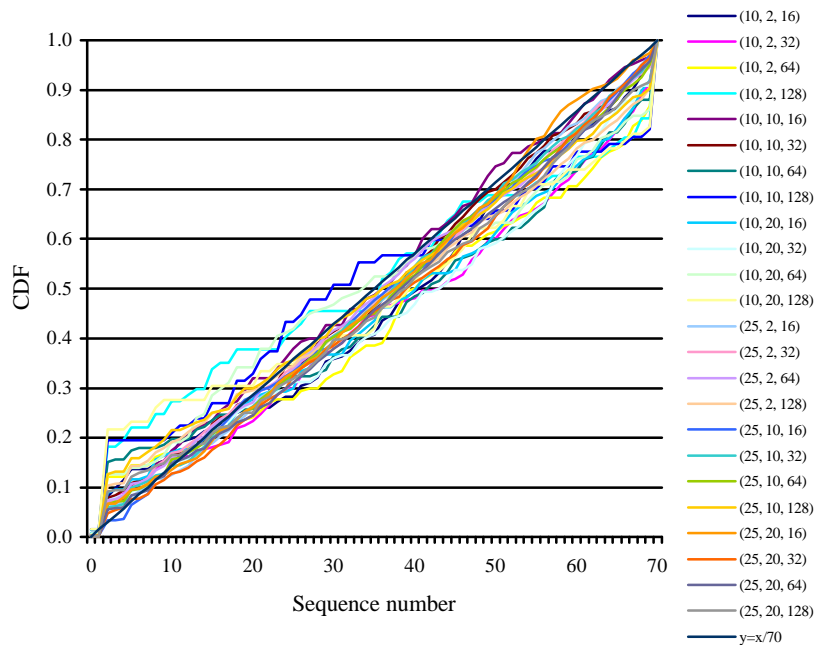


Fig. 2. Cumulative distribution function (*CDF*) of ECN marks across a number of experiments. Experiment configuration parameters are given as (number of clients, d in ms, QL at router B in packets). See also TABLE 1

It should be noted that very little in the simulation is driven by random numbers: ns-2 runs actual TCP code, and packet transfer times are deterministic for a given link, once its transmission rate and propagation delay are defined. The only aspect of

the simulation that receives generated random variate input is the RED marking at the router B buffer. Because of this, file transfer averages of the kind we report on would not vary significantly across independent replications of the experiment, and the results we present, based on a single replication, are quite representative (representative, that is, for the particular case study network configuration we are simulating). In the few cases where we implemented an independent replications procedure, samples based on 12 replications yielded small variance. The 98% level confidence intervals for single-file transfer times, for example, were mostly within $\pm 5\%$ of the sample means.

Of course, a single replication’s $11n$ sample values are not mutually independent since they result from the interaction of n parallel sets of serial TCP transfers competing for network resources. They are not even identically distributed since, as each client completes its 11 file transfers and “shuts down”, the remaining clients experience less competition for network resources. As such, we have not attempted to formally wrap the (single-run) averages reported in confidence intervals.

3.2 Parameter Setting

Our experiments are primarily aimed at comparing the performance of standard TCP (over DT and RED) with that of TCP/ECN. The number of clients ranges from 5 to 25, and the buffer size (QL) at router B from 8 to 128 packets. RED parameters [7] were set as follows: the minimum (\min_{th}) and maximum (\max_{th}) thresholds varied depending on QL, see Table 1; 10% for the maximum dropping probability, \max_p , for all tests presented in this paper; and a weighting factor, w_q , used to calculate the average queue size, fixed at 0.002 for all tests. RED was always configured with the “gentle” option [10].

Table 1. RED parameter settings

QL	16	32	64	128
\min_{th}	5	10	20	40
\max_{th}	15	30	60	120

4 Results and discussion

Our primary interest in this paper is to compare TCP when complemented by a “pure” marking mechanism to the more usual situation of “standard” TCP with a dropping mechanism. The former is represented by TCP/ECN, and the latter by TCP over DT. TCP over RED, as such, is ancillary to our focus because it still uses dropping as a congestion notification mechanism. ECN is dependent on an AQM mechanism, in this case RED’s. As such, results from TCP over RED provide a “control” group which enable us to apportion the gains achieved by TCP/ECN between, on the one hand, its AQM scheme and, on the other, the marking mechanism per se. Due to space limitations, however, we only present results for 10 and 25 clients. Results

with fewer clients are different to the extent that they entail lower levels of congestion, and will be briefly discussed below. Results with more clients are similar with the ones we present in this paper, if QL is increased. It should be noted, however, that with 10 clients the router B buffer operated in the RED region most, but not all, of the time, and with 25 clients virtually all of the time.

4.1 Transfer Duration

In contrast to previously published studies [2][6][13][14], our experiments produced results that consistently show no significant improvement in TCP goodput, across different scenarios, when ECN is employed. In fact, in some cases the average transfer times to complete the 100 KB download are somewhat larger (and hence the average goodput smaller) for TCP/ECN. For example, Fig. 3 presents the average transfer times (and the standard deviations) for 10 and 25 clients, with propagation delay d along the bottleneck equal to 20 ms, for four varied QLs.

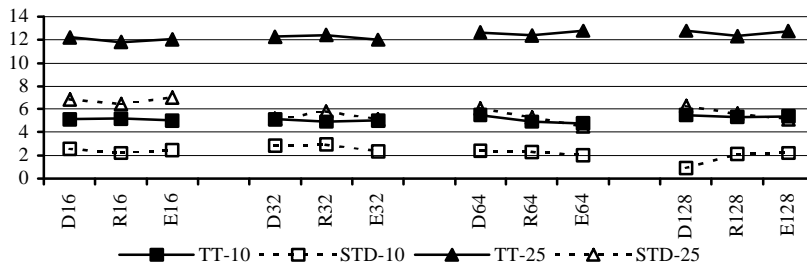


Fig. 3. Average transfer duration (TT) and standard deviation (STD), in seconds, with 10 and 25 clients competing ($d = 20$ ms). (D : DT; R : RED; E : ECN)

At first sight, it seemed intriguing that TCP/ECN does not lead to improved performance. Closer examination of the experiment data revealed that TCP/ECN indeed does become aware of congestion conditions sooner than does standard TCP and, therefore, backs off in a more timely manner (see subsection 4.2, below). However, it is unable to detect improved network conditions any more efficiently than standard TCP since, in this respect, both implement the same mechanism for congestion window expansion. Earlier studies have shown that, when TCP/ECN competes with ECN-unaware traffic (*i.e.*, over RED), it has the advantage in detecting congestion and so avoids transmitting in times of congestion. This in turn helps it avoid packet drops, and thus prevents TCP's congestion window collapse, yielding increased goodput. On the other hand, when all traffic is TCP/ECN none of the senders has a competitive advantage and "the gains of one flow are the losses of another". For any given QL, TCP over DT undertakes more segment transmissions overall in order to complete the 11n transfers than TCP/ECN (see subsection 4.2, below). With less timely information about congestion, TCP over DT attempts some segment transmis-

sions that TCP/ECN does not (since the latter halves its congestion window immediately upon receipt of ECNs). Some of these “risky” (because of rising congestion levels) transmissions succeed, allowing TCP to achieve on the average similar goodput with TCP/ECN.

Furthermore, TCP’s highly tuned algorithms, combined with flow multiplexing, yield bottleneck link utilization in the mid- to high ninety percent range (especially when 25 clients are involved), leaving very limited spare bandwidth, which causes the average transfer times to vary hardly at all across varying QLs. Note that these QLs are sufficiently large to ensure that bottleneck link starvation does not occur.

4.2 Network and Receiver Overhead

We define network overhead as the number of excess packets that the TCP sender transmits for the specified application payload (100 KB), over and above the minimum required in an uncongested network. Network overhead is expressed as a percentage of this minimum, and includes packets that are dropped at the bottleneck router, packets corrupted in transmission, as well as duplicate (*i.e.* unnecessarily retransmitted) packets. In contrast, receiver overhead is defined as the number of packets received at the client in excess of the application payload minimum, also expressed as a percentage of the latter. Thus, receiver overhead includes duplicate and corrupted packets, but not packets dropped at router B. Both overheads are presented in Fig. 4 below for the case of 10 and 25 clients, with $d = 20$ ms.

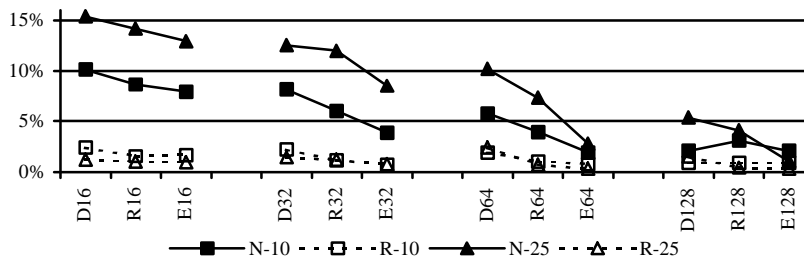


Fig. 4. Network (N) and receiver (R) overhead with 10 and 25 clients competing ($d = 20$ ms). (D : DT; R : RED; E : ECN)

Taking network overhead first, and as can be seen from the figure, TCP/ECN induces smaller overhead than TCP does. Consider, for example, the case of $QL = 64$. The TCP network overhead is 10.18% over DT, and 7.32% over RED; TCP/ECN’s is 2.78%. That is, network overhead is reduced by as much as 72%, yielding 7.40% fewer total transmitted packets. Also note that in this particular case, TCP/ECN absolute gains (in terms of number of packets) increase as more clients compete, while relative network overhead (in terms of percentage) remains constant. With 10 clients active, TCP over DT transmits 445 packets more packets than what is re-

quired to transfer the total application payload; if TCP/ECN is used, this figure drops to 149 packets (*i.e.*, network overhead reduction of 66%). With 25 clients, the figures are 1960 vs. 535 packets, yielding a reduction of 72%.

Clearly, not all gains are due purely to the marking mechanism: RED itself achieves some gains over DT. The only exception is for $QL = 128$ with 10 active clients, where router B buffer can accommodate all flows with hardly any drops under DT. There are almost 13 slots in the buffer for each active flow, each of which experiences a negligible number of congestion incidents. With RED, random segment losses are induced, yielding poorer network overhead. TCP/ECN's network overhead is not affected since packets are marked rather than dropped.

Moreover, ECN yields the same or less network overhead than DT with twice the buffer size. This resource efficiency can lead to improved services. Our results confirm that, as expected in a highly congested network, the average packet transfer time is halved when QL is halved. Although interactive flows are not dealt with in this paper, the potential gains for short, transactional requests are clear.

In general, when QL is under-provisioned RED degenerates and causes more packet drops than DT, due to occasional random dropping over and above buffer overflow [5]. On the other hand, when QL is over-provisioned the network overhead is similar for ECN and DT, and slightly better than for RED with dropping. In terms of receiver overhead TCP/ECN consistently improves on TCP over DT, if only modestly: 2% of total packets received, though this represents a relative gain of as much as 88% (Fig. 4).

Of course, when congestion levels are lower, the benefits of ECN are negligible. For example, with five clients, no background traffic, and d small (2 ms), the bottleneck router does not operate in the RED region, yielding rather disappointing results in terms of network overhead for TCP/ECN (not shown). Increasing d to 20 ms, we have TCP/ECN yielding a slight improvement in network overhead.

4.3 Power Efficiency

The resource efficiency of TCP/ECN can be of greater importance in networks other than the traditional wired Internet. In particular, resource efficiency can translate into power savings for battery-operated (mobile) hosts. Although exact communications-related power consumption depends on the hardware and software specifics of the battery-operated device, it is important to use a transport protocol that is as resource efficient as possible.

With the proliferation of wireless networks, TCP is being called on to serve as the transport protocol in hybrid wired/wireless environments. It is well established in the literature that TCP Reno does not perform well in such environments, because, amongst other reasons, TCP interprets all segment losses as congestion incidents [11]. ECN can provide more information to a TCP sender about the congestion levels and therefore allow it to make more intelligent decisions. In order to see the effect of rather infrequent random losses on the performance of TCP/ECN, and compare it with the results presented already, we repeated the experiments with the addition of a

rather simple, Bernoulli trials error model on the links connecting the clients with router A, *i.e.* the clients' access subnetwork.

In any battery-operated device, three factors determine communications-related power consumption: (a) the amount of packets transmitted, (b) the amount of packets received, and (c) the amount of time spent idling. In our case study, assuming that clients are battery-operated: (a) corresponds to the ACKs sent back to the server; (b) to packets received; and (c) can be associated in a straightforward manner with the transfer time. Therefore, the metrics used in the previous section can serve as rough indicators for the energy efficiency of TCP/ECN.

4.4 Receiver Overhead with Random Errors

Receiver overhead can provide us with a metric of totally wasted energy because it measures the amount of packets that are redundant. Fig. 5 presents network and receiver overhead when 1%-probability random errors are introduced in the clients' access subnetwork, and d is set to 2 and 20 ms, respectively. TCP/ECN achieves smaller receiver overhead across all scenarios. For example, for 25 active clients, and $QL = 128$, TCP receiver overhead is 3.07% for DT, 0.87% for RED, and 0.56% for ECN. In other words, the reduction in receiver overhead for TCP/ECN is almost 82%; this translates to 2.5% fewer total packets received.

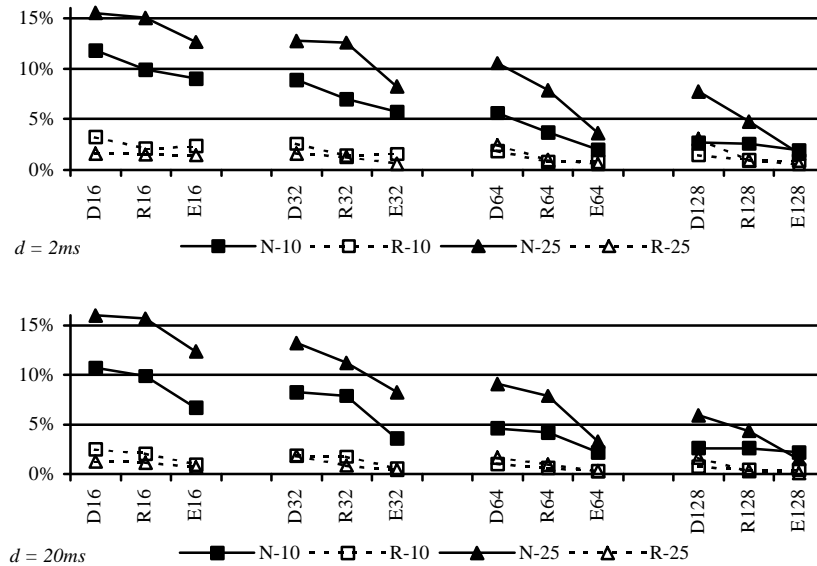


Fig. 5. Network (N) and receiver (R) overhead (10 and 25 clients); random errors are introduced. (D: DT; R: RED; E: ECN)

Light random errors force TCP to behave more conservatively and so impose slightly smaller load on the network. Comparing the results presented in this section with those in subsection 4.2, we conclude that TCP/ECN still maintains an advantage, in both network and receiver overhead.

4.5 Transfer Duration with Random Errors

From an energy point of view, and all other things being equal, the smaller the amount of time needed to transfer the application payload the larger the savings. Fig. 6 illustrates the average transfer time and the standard deviation when random errors are introduced, for d set to 2 and 20 ms, respectively. The results are similar with the ones presented in subsection 4.1, above. There is hardly any variation in average transfer times across all scenarios when only 10 clients are active. With 25 clients active, TCP/ECN achieves slightly improved transfer times in some scenarios (*e.g.*, $QL = 16$), and worse ones in others (*e.g.*, $QL = 64$, $d = 2ms$). In general, though, TCP/ECN does not have a competitive advantage over standard TCP, and the arguments presented in subsection 4.1 apply here too. Thus, if we consider transfer times alone, TCP/ECN is no more power-efficient than TCP over DT.

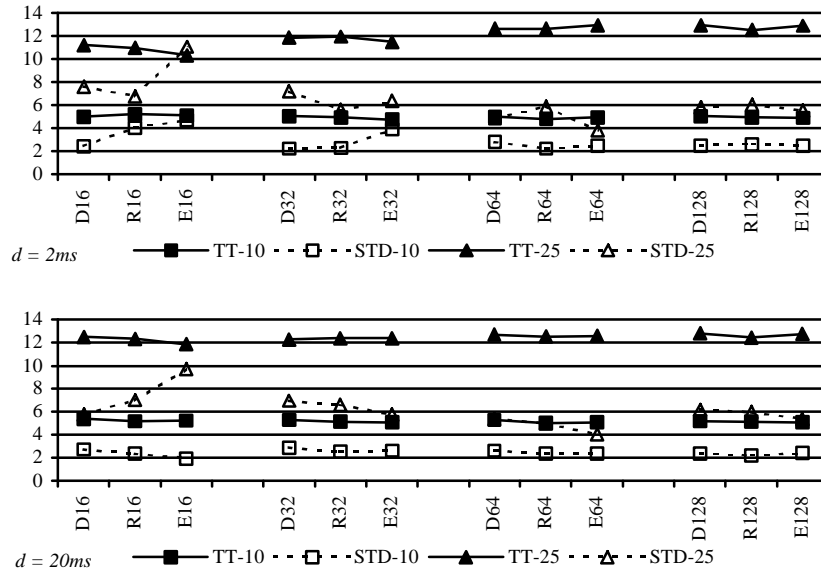


Fig. 6. Average transfer duration (TT) and standard deviation (STD), in seconds, (10 and 25 clients); random errors are introduced. (D: DT; R: RED; E: ECN)

4.6 Transmission Energy Savings

The use of the delayed ACKs algorithms is very popular with most TCP implementations and has been shown to be network resource efficient. It is also deemed an efficient power-conserving mechanism. However, delayed ACKs double the amount of time that a TCP sender spends in Slow Start, which means that transfer times, especially for short files, are prolonged. In light of the results presented in this paper, we want to investigate the impact of delayed ACKs on TCP/ECN.

Our experiments assume that battery-operated clients have no data to send and always act as receivers. However, they do initiate file transfers and transmit TCP ACKs. Though ACKs are small in size (40 bytes for the IP and TCP header, plus the headers of the lower levels), they can be as costly to transmit as full sized segments. Balakrishnan *et al.* [3] note that in many asymmetric networks, such as wireless packet radio, MAC schemes introduce overhead per upstream transmission, which can make the transmitting of short packets (including TCP ACKs) as costly as transmitting MTU-sized packets. Therefore, limiting the number of ACKs that the receiver sends can translate into significant power savings.

Fig. 7 presents the average number of ACKs generated under a variety of scenarios. In all these scenarios, TCP/ECN generates fewer ACKs than TCP over DT. Moreover, TCP/ECN generates fewer ACKs than TCP over RED, except for $QL = 32$, $d = 2$ ms, and 10 active clients. The maximum gains for TCP/ECN are achieved when $QL = 128$, $d = 2$ ms, and 25 clients are active: TCP over DT generates almost 54, while TCP/ECN generates approximately 44, ACKs, a reduction of 15%. Notice that part of these gains is apportioned to using RED at the bottleneck router. If QL is reduced to 64 packets, TCP/ECN clients generate (on the average) 10% fewer ACKs. Further reductions in QL translate into smaller gains for TCP/ECN.

5 Conclusion

In this paper we use simulation to study the case of moderately-short TCP file downloads with a bottleneck link in the end-to-end path. Our focus is on the resource efficiency of an all-TCP/ECN environment, as compared to all-DT and all-RED environments.

We observe, contrary to our expectations, that TCP/ECN does not achieve improved goodput. On a more positive note, we show that ECN leads to significant gains in network overhead, and modest, but measurable, gains in receiver overhead. A rough rule of thumb is that an all-TCP/ECN environment maintains the same overall performance, in terms of transmission overhead for duplicate and dropped packets, and for ACKs, as an all-DT environment with twice the buffer size at the bottleneck router.

Introducing light, random, wireless-like error conditions in the receivers' access network does not degrade ECN's resource efficiency. This suggests promising power-conserving potential for battery-operated mobile devices, especially in limiting the number of generated ACKs.

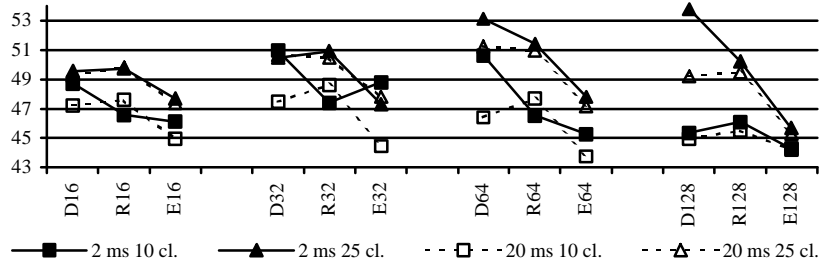


Fig. 7. Average number of ACKs sent when random errors are introduced in the clients' (*cl.*) access subnetwork. (*D*: DT; *R*: RED; *E*: ECN)

References

1. Allman, M., Paxson, V. and Stevens, W. R.: TCP Congestion Control. RFC 2581, April 1999.
2. Athuraliya, S., Li, V., Low, S., and Yin, Q.: REM: Active Queue Management. In: IEEE Network, May/June 2001.
3. Balakrishnan, H., *et al.*: TCP Performance Implications of Network Asymmetry. Internet Draft, <<http://www.ietf.org/html.charters/pilc-charter.html>>, (March 2002).
4. Braden, B., *et al.*: Recommendations on Queue Management and Congestion Avoidance. RFC 2309, April 1998.
5. Christiansen, M., Jeffay, K., Ott, D., and Smith, F.D.: Tuning RED for Web Traffic. In: Proceedings of ACM SIGCOMM 2000, Stockholm, Sweden, August 2000.
6. Floyd, S.: TCP and Explicit Congestion Notification. In: ACM Computer Communication Review, Vol. 24, No. 5, October 1994.
7. Floyd, S. and Jacobson, V.: Random Early Detection Gateways for Congestion Avoidance. In: ACM/IEEE Transactions on Networking, Vol. 3, No. 1, August 1993.
8. Lavens, K., Key, P., and McAuley, D.: An ECN-based end-to-end congestion-control framework: experiments and evaluation. Microsoft Research Technical Report, MSR-TR-2000-104, October 2000.
9. May, M., Bolot, J., Diot, C., and Lyles, B.: Reasons Not to Deploy RED. In: Proc. of 7th International Workshop on Quality of Service (IWQoS'99), London, UK, June 1999.
10. UCB/LBNL/VINT Network Simulator - ns (version 2). <<http://www.isi.edu/nsnam/ns>>, (March 2002).
11. K. Pentikousis: TCP in wired-cum-wireless environments. In: IEEE Communications Surveys, Vol. 3, No. 4, Fourth Quarter 2000.
12. Ramakrishnan, K.K., Floyd, S., and Black, D.: The Addition of Explicit Congestion Notification (ECN) to IP. RFC 3168, September 2001.
13. Slim, J. H., and Ahmed, U.: Performance Evaluation of Explicit Congestion Notification (ECN) in IP Networks. RFC 2884, July 2000.
14. Zhang, Y. and Qiu, L.: Understanding the End-to-End Impact of RED in a Heterogeneous Environment. Cornell CS Technical Report 2000-1802, July 2000.