Enabling a Bufferless Core Optical Network Using Edge-to-Edge Packet-Level FEC

Arun Vishwanath, *Member, IEEE,* Vijay Sivaraman, *Member, IEEE,*
Marina Thottan, *Member, IEEE* and Constantine Dovrolis, *Member, IEEE*

***Abstract*—To cope with the phenomenal growth of the Internet over the next decade, core networks are expected to scale to capacities of terabits-per-second and beyond. Increasing the role of optics for switching and transmission inside the core network seems to be the most promising way forward to accomplish this capacity scaling. Unfortunately, unlike electronic memory, it remains a formidable challenge to build even a few packets of integrated all-optical buffers. In this context, we envision a bufferless (or near-zero buffer) core optical network and make three contributions: First, we propose a novel edge-to-edge based packet-level forward error correction (FEC) scheme that combats packet loss in the bufferless core, and characterise the impact of FEC strength on loss at a single link. Second, we develop a global optimisation framework for multi-hop networks, and propose a heuristic algorithm that adjusts FEC strength to achieve fairness amongst the different single- and multi-hop flows. Finally, we evaluate the performance of our FEC scheme for realistic mixes of short- and long-lived TCP flows, and show that edge-to-edge packet-level FEC can be tuned to effectively mitigate contention losses in the core, thus opening the doors to bufferless optical networks in the near future.**

***Index Terms*—all-optical core network, TCP, packet-level FEC, bufferless networks, fairness**

I. Introduction

THE Internet has witnessed tremendous growth over the past twenty years, both in terms of user traffic and core link capacities. Current Internet traffic is already in the exabytes/year, and projections show global IP traffic will reach zettabytes in the next five years [1]. This has led to a significant increase in the power/energy requirements of the associated networking infrastructure. It is clear that the density of power consumption is highest at core routers as they are being scaled to switch terabits-per-second of bandwidth to support the increased traffic demand.

Packet buffers (SRAM and DRAM) are an integral part of every router/switch as they absorb transient bursts of traffic, and thus play a fundamental role in keeping packet loss to a minimum. On the downside, they introduce delay and jitter, and are largely responsible for the high cost, power consumption and heat dissipation in core routers [2]–[4]. This has led high capacity router/switch manufacturers and network providers to consider optical data switching in core routers [5]. For example, recent work has demonstrated prototypes of all-optical packet switched routers: IRIS (Integrated Router Interconnected Spectrally) [6] and LASOR [7], which employ integrated optical buffers [8], [9]. However, incorporating even a few packets of buffering using an all-optical on-chip memory is a formidable challenge due to the inherent complexity associated with maintaining the quality of the optical signal and the physical size limitations of the chip [9]. At the moment, our IRIS router is capable of buffering 100 ns worth of data. At 40 Gbps line rate, this translates to 500 bytes, roughly equivalent to one average Internet packet [10].

Given that incorporating all-optical buffering into commercial routers is likely to remain a veritable challenge for years to come, in this paper we investigate if a high-speed wide-area bufferless (or near-zero buffer) core optical network can deliver acceptable end-to-end performance. While [12] has also considered bufferless optical networks, it assumes wavelength-conversion capability, which incurs significant expense. Other studies on router buffer sizing (surveyed in our recent article [11]) have considered buffers of size ranging from a few tens to thousands of packets, which are not amenable for all-optical implementation. Nevertheless, these studies have applied techniques such as traffic conditioning (for example our traffic pacing mechanism proposed in [13]) or new TCP variants (such as LT-TCP [14]) to mitigate or adapt to losses in the network. While these approaches have their merits, in this paper we propose a new technique that applies packet-level forward error correction (FEC) coding at the electronic edges to recover packets lost in the bufferless core. In this context we make the following three new contributions.

1) First, we propose the edge-to-edge packet-level FEC mechanism, and analytically deduce the optimal FEC strength for flows that share a single core link. The analysis is corroborated against simulation and shows that packet-level FEC can help TCP achieve good throughput over a bufferless core link.
2) Second, we consider a representative core network topology and show that multi-hop TCP flows can have significantly worse performance than single-hop flows. To address this unfairness, we develop a global optimisation framework that allows determination of FEC strength on a per-flow basis to achieve max-min fairness. We also propose a practical heuristic that achieves good fairness by choosing the FEC strength based on estimated edge-to-edge loss rates in the network.

A. Vishwanath is with the Centre for Energy-Efficient Telecommunications, University of Melbourne; arun.v@unimelb.edu.au.

V. Sivaraman is with the School of EE&T, University of New South Wales, Sydney, Australia; vijay@unsw.edu.au.

M. Thottan is with the Networking Research Lab, Bell-Labs Alcatel Lucent, USA; marinat@alcatel-lucent.com.

C. Dovrolis is with the College of Computing, Georgia Institute of Technology, USA; dovrolis@cc.gatech.edu.

This submission is an extended and improved version of our paper presented at the IEEE INFOCOM 2010 conference [30].

3) Finally, we study the efficacy of FEC for realistic mixes of short-lived and long-lived TCP flows and show that packet-level FEC, when tuned properly, can be very effective in combating high core losses, thus helping overcome a major obstacle for the realisation of all-optical bufferless core networks in the future.

The rest of the paper is organised as follows. In Section II, we describe our edge-to-edge packet-level FEC scheme. Section III motivates our choice of using FEC, while Section IV analyses TCP performance in a single link topology and highlights unfairness in a multi-link network. We develop a framework for optimising fairness in Section V and evaluate a practical heuristic algorithm for a representative wide-area network. Section VI concludes the paper.

II. Edge-to-Edge Packet-Level FEC Framework

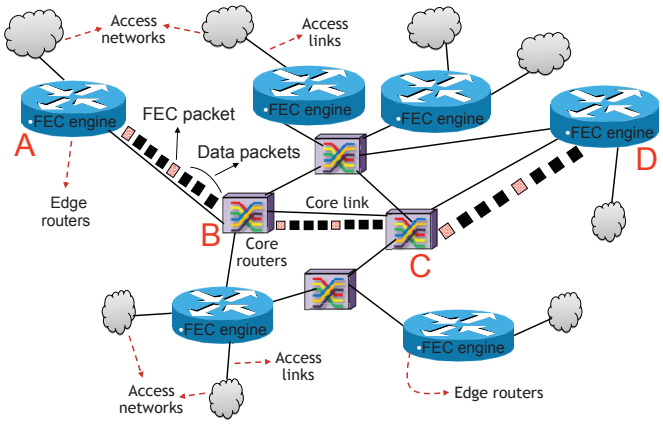

Fig. 1. Topology to illustrate our edge-to-edge packet-level FEC framework

Fig. 1 shows a small segment of a typical ISP network comprising of optical core routers and electronic edge routers. The distinguishing feature between the core and edge routers is that the core router links are bufferless (near-zero buffer) while the electronic router links have large buffers. FEC is performed on the aggregate traffic flowing from an ingress to an egress edge router, and not at the granularity of individual TCP flows. As an example, all traffic entering the edge router in say New York city and exiting at the edge router in say Los Angeles constitutes an edge-to-edge flow, and is protected by its own FEC. This FEC scheme is scalable since an edge router need only maintain FEC state for $N-1$ edge-to-edge flows, where N is the number of edge routers in the network. Further, the FEC framework is entirely controlled by the ISP, and is completely transparent to the end-hosts.

An ingress edge router receives traffic on its access links (DSL, cable modem, etc.), classifies packets to the appropriate edge-to-edge flow, and performs FEC computations as follows. An FEC packet is computed by bit-wise XOR-ing a block of k successive data packets (smaller packets can be assumed to be padded to the maximum packet size with zeroes). The number k, henceforth referred to as block-size, denotes the strength of the FEC scheme. The ingress edge therefore transmits one FEC packet for every block of k data packets, allowing the corresponding egress router to recover from loss of at most one data packet from the block. We choose this XOR based FEC scheme primarily for its simplicity and ease of implementation, noting its real-time performance and low computation/storage requirements. We also note that Akamai has reported [15] trialling such a packet-level 'parity based' scheme on their content delivery network for transfer of real-time traffic streams. More sophisticated schemes such as Reed-Solomon codes or Fountain codes are left for future work.

To illustrate the operation of the FEC scheme, the figure shows traffic flowing from edge router A to edge router D along the path A-B-C-D. Assuming block-size is three, A keeps a running XOR of data packets destined to D. After every third packet, A transmits the FEC packet comprising the XOR value and clears the XOR. D in turn maintains a running XOR of packets it receives from A. If it receives all but one data packet in the window of four packets (known via a sequence number in the packet header), the running XOR is XOR-ed with the FEC packet to recover the lost data packet, which is then forwarded on, and the XOR is cleared.

The FEC strength in terms of block-size k determines the bandwidth overheads of the scheme. The inflation in bandwidth by fraction $1/k$ increases with decreasing k (i.e. stronger FEC), while the ability to recover from lost packets also increases. This price to eliminate buffering may be worth paying if the optical core has abundant bandwidth, as investigated in subsequent sections.

III. Motivation for Choosing FEC

We envision a core network (such as Fig. 1) where core links are fed by many edge links (which in turn are fed by several access links), and core link capacities are larger than the edge/access links feeding them (i.e. the edge/access links are the bottleneck). In such a network, losses will occur in the core not because a single edge-to-edge flow can saturate the core link with a burst of back-to-back packets, but because of the simultaneous arrival of packets from different links (belonging to different edge-to-edge flows) at a given time. Therefore, loss perceived by a single edge-to-edge flow would be random, contention based losses, and not bursty losses.

To verify this hypothesis we simulated over 3000 edge-to-edge TCP flows on the NSFNet core network topology shown in Fig. 2 (described in detail in Section IV-C). In Fig. 3 we plot the complementary cumulative distribution function (CCDF) on log scale of the number of packets received by an egress router between two successive packet drops for a randomly picked 2-hop flow (similar plots were obtained for other randomly picked 1- and 3-hop edge-to-edge flows as well, not shown for brevity). We observe that the CCDF plot can be approximated as a straight line, meaning that the distribution of the number of packets received between two successive drops follows a geometric distribution. In other words, loss process is memoryless, akin to tossing an independent coin for each packet to decide if is lost or not.

To further validate our hypothesis, we use the Wald-Wolfowitz runs test [17] operating as follows. For every edge-to-edge flow in the above simulation, we count the number of "runs", where a run constitutes an uninterrupted sequence of

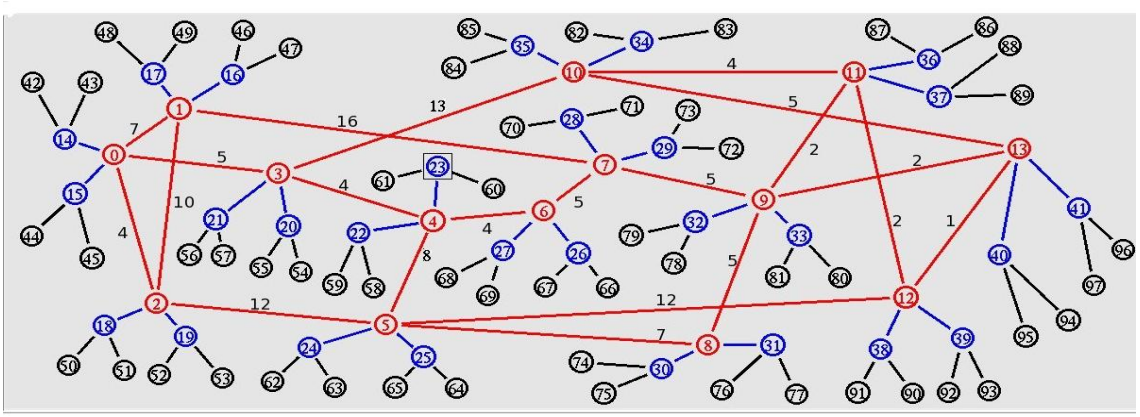

Fig. 2. Example NSFNet topology with 2 access links per edge node and 2 edge links per core node from *ns-2* network animator

packets that are either received by the egress router or dropped in the core. As an example, the sequence "RRDRRRDDR" has five runs, three are due to packets received (i.e. R) and two are due to packets dropped (i.e. D).

Our null hypothesis H_0 can be expressed as "packet losses for the edge-to-edge flow occur randomly", while the alternate hypothesis H_a is "packet losses for the edge-to-edge flow do not occur randomly". Considering any particular edge-to-edge flow, if W, n_r and n_d denote respectively the number of runs, the number of packets received by the egress router and the number of packets dropped in the core network, and if n_r and n_d are greater than ten, then under the null hypothesis H_0, $W \sim N(\mu, \sigma^2)$ where $\mu = 1 + \frac{2n_r n_d}{n_r+n_d}$ and $\sigma = \frac{(\mu-1)(\mu-2)}{n_r+n_d-1}$ (i.e. W follows the given Normal distribution with mean μ and variance σ^2) [17]. Thus, our standard normal test statistic Z can be expressed as $Z = \frac{W-\mu}{\sigma}$. Once Z is known, we can easily obtain the corresponding p-Value (for the two-tailed test) from the standard normal distribution table. If α represents the significance level of the test, then we can reject H_0 in favour of H_a if the corresponding p-Value is less than or equal to α.

We obtained the CDF of the p-Values when the Wald-Wolfowitz runs test was carried out on every edge-to-edge flow. For a typical $\alpha = 0.05$ we note that H_0 is rejected in favour of H_a for only about 11% of the flows, while for the remaining 89% of the flows, there is insufficient evidence to reject the null hypothesis that packet losses occur randomly.

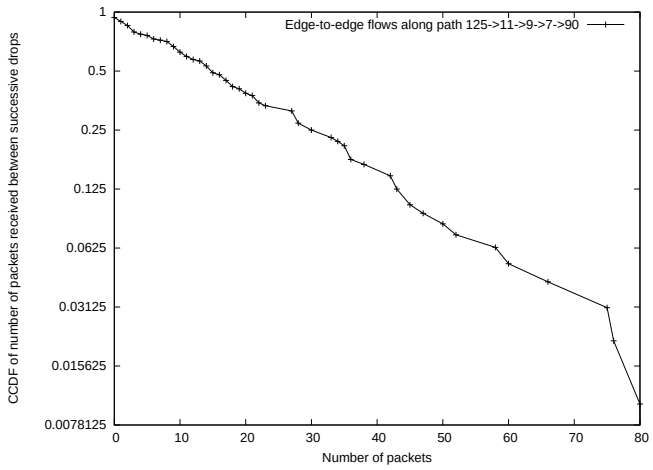

Fig. 3. CCDF of the number of received packets between two consecutive dropped packets for an example 2-hop flow

In summary, Fig. 3 and the hypothesis test confirm our intuition that when core link capacities are higher than the access/edge link capacities, loss for an edge-to-edge flow will occur randomly in the core. FEC is known to be effective in recovery when losses are random (rather than bursty), motivating our choice of using FEC to recover from packet loss in the bufferless core.

Secondary reasons for choosing FEC are that it is a well established technique, and can be easily implemented in hardware. FEC can introduce some bandwidth overhead, but this is a small price to pay for building power efficient bufferless core optical routers [18], particularly so because ISPs typically operate their core networks at relatively low loads ($\approx$ 15-25%) [19]. In addition, packet-level FEC has, to the best of our knowledge, not been studied before in the context of enabling a bufferless core network. Finally, our scheme can easily co-exist with other techniques outlined above to reduce loss, such as traffic shaping/pacing, etc.

IV. Configuring FEC for a Single Link Topology

In this section, we study via analysis and simulation the FEC strength for optimal loss/goodput performance for single-hop TCP flows sharing a core link, and highlight the fairness issues that arise when extending the framework to multi-hop networks.

A. Goodput for TCP Flows

We begin by implementing the above edge-to-edge FEC framework in *ns-2* (version 2.33), and apply it to the single core link dumbbell topology shown in Fig. 4 (the efficacy of FEC in a wide-area mesh network topology with thousands of TCP flows is studied in subsequent sections). Ten edge links feed traffic into the single buffer core link at router C_0, with each edge link in turn fed by three access links. The 30 end-hosts each have 5 TCP (Reno) agents, and the network therefore simulates 150 long-lived TCP flows (short-lived TCP flows are considered in Section V). Similarly the TCP flows

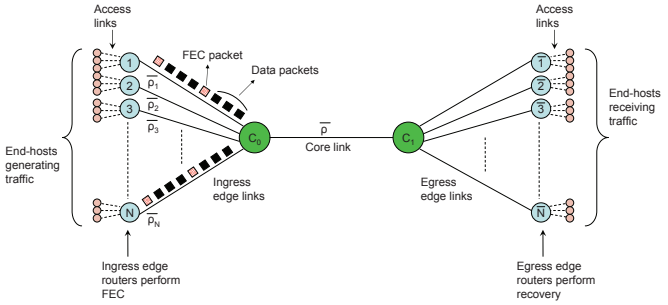

Fig. 4. Example dumbbell topology with a single core link

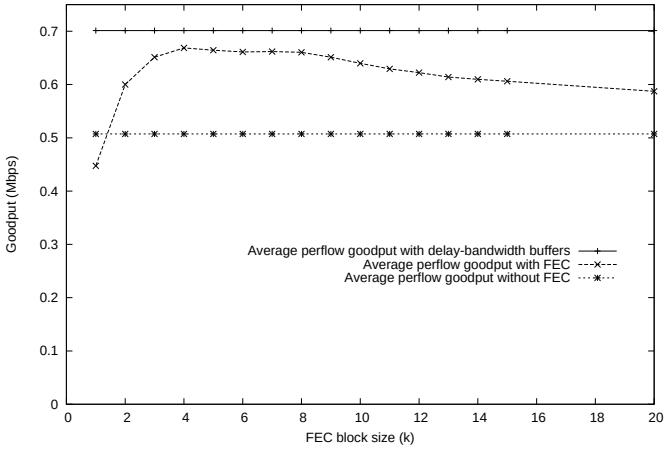

Fig. 5. Average perflow goodput for 150 TCP flows on dumbbell topology

are sinked by the 30 end-hosts on the right. The propagation delays on the access and edge links are uniformly distributed between $[1, 5]$ ms and $[5, 15]$ ms respectively, while the core link C_0-C_1 has delay 30 ms. The access link transmission rates are uniformly distributed in $[3, 5]$ Mbps, all edge links operate at 40 Mbps, and the core link at 400 Mbps. For these link speeds it can be seen that the access link is the bottleneck, since each flow's fair-share of the bandwidth on the access links varies between 0.6-1 Mbps, while on the edges and the core it is 2.67 Mbps. The start time of the TCP flows is uniformly distributed in the interval $[0, 10]$ sec and the simulation is run for 35 sec. Data in the interval $[20, 35]$ sec is used in all our computations so as to capture the steady-state behaviour of the network.

We measure the average per-flow TCP goodput for each setting of the FEC block-size k in simulation. We use goodput as a metric since it has been argued to be the most important measure for end-users [20], who want their transactions to complete as fast as possible. Our simulations use a buffer size of 1 kB to accommodate a single TCP packet (TCP packets in our simulation are of size 1 kB) that is stored and forwarded by the core router.

Fig. 5 shows the average per-flow TCP goodput as a function of block-size k. For comparison, it also depicts, via horizontal lines, the average goodput without FEC (the bottom line), and the average goodput if the core link was to have sufficient (delay-bandwidth) buffering of around 12.5 MB (top line). Large buffers yield a per-flow goodput of 0.7 Mbps, while eliminating buffers reduces this goodput to 0.5 Mbps, a sacrifice in goodput of nearly 30%. Employing edge-to-edge FEC over the near-zero buffer link can improve per-flow goodput substantially, peaking at nearly 0.68 Mbps when the FEC block-size k is in the range of 3-6, and bringing the per-flow TCP goodput for the link to within 3% of a fully-buffered link. This small sacrifice in goodput is a worthy price to pay for eliminating buffering at router C_0.

Another interesting aspect to note from Fig. 5 is that TCP goodput initially increases with FEC block-size k, reaches a peak, and then falls as k increases. Qualitatively, this is because stronger FEC (i.e. smaller block-size k) in general improves the ability to recover from loss, but is also a contributor to loss since it increases the load on the link by introducing redundant packets. In the next subsection, we capture this effect via a simple analytical model to determine the optimal setting of FEC block-size that minimises loss.

B. Analytical Model for Loss Minimisation with FEC

We develop a simple approximate analytical model to quantitatively understand the impact of FEC strength on edge-to-edge loss, and to identify the block-size settings that achieve low loss and consequently larger goodput. Our analysis relies on several simplifying assumptions:

1) Traffic entering the core links from various edge links are independent of one another. This is a reasonable assumption, even for TCP traffic, when the number of flows is large enough [2].

2) Packets arrive at the core router according to a Poisson process. This is a valid assumption in our case because all the traffic that enters the core arrive from flows that are of much lower rate (i.e. bottlenecked at the access/edge) when compared to the capacity of core links [21]. Also, each TCP flows's window will be quite small (since core links have only a single packet buffer), implying that each flow will only generate a small amount of traffic per RTT, the aggregation of such a large number of independent flows can reasonably be assumed to be Poisson [22].

3) We do not model feedback, i.e. TCP's adjustment of rate in response to loss. Instead, we assume that the steady-state load of a link is determined by the arrival rate of new flows and the average flow size on that link.

Denote by ρ_i the original load (i.e. load without FEC) on each of the edge links (i, C_0), $i \in \{1, N\}$ (see Fig. 4). The offered load at the core link C_0-C_1 is then $\rho = \sum_{i=1}^{N} \rho_i$. Now, if each edge link performs FEC using block-size k, the new load $\overline{\rho_i}$ on each of these edge links is

$$\overline{\rho_i} = \left(\frac{k+1}{k}\right) \rho_i \tag{1}$$

since FEC inserts one additional packet for every k data packets. Correspondingly, the offered load $\overline{\rho}$ post-FEC at the core link is

$$\overline{\rho} = \sum_{i=1}^{N} \overline{\rho_i} = \sum_{i=1}^{N} \left(\frac{k+1}{k}\right) \rho_i = \left(\frac{k+1}{k}\right) \rho \tag{2}$$

We can now model the core link as a simple M/M/1/B queue with the state representing the number of packets in the system. In our case we have three states - corresponding

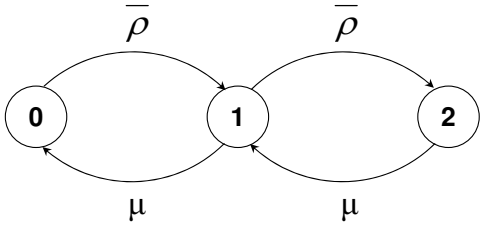

Fig. 6. M/M/1/B Markov chain model of a core link with buffer size one

to an empty system (state 0), one packet in the server (state 1) and one packet each in the buffer and the server (state 2). The resulting Markov chain is shown in Fig. 6. In such a system, loss occurs iff an arriving packet finds a full system, which is nothing but the probability of the system being in state 2. By normalising the service rate μ to be unity, the loss probability $\mathbb{L}_c$ in the core is obtained from the loss probability of an M/M/1/2 system [23], which is then given by

$$\mathbb{L}_c = \frac{\overline{\rho}^2}{1 + \overline{\rho} + \overline{\rho}^2} \tag{3}$$

Knowing the probability of packet loss in the core, we can now estimate the edge-to-edge packet loss probability $\mathbb{L}_e$ by computing the expected number of irrecoverably lost packets in a window of $k+1$ packets (comprising k data packets and one FEC packet) as follows:

$$\mathbb{L}_e = \mathbb{L}_c \sum_{j=1}^{k} \binom{k}{j} (\mathbb{L}_c)^j (1-\mathbb{L}_c)^{k-j} \frac{j}{k} + (1-\mathbb{L}_c) \sum_{j=2}^{k} \binom{k}{j} (\mathbb{L}_c)^j (1-\mathbb{L}_c)^{k-j} \frac{j}{k} \tag{4}$$

The first term on the right in Eq. (4) captures the case when the FEC packet is lost along with j data packets, in which all j data packets are irrecoverable, while the second term captures the case when the FEC packet arrives and $j \geq 2$ data packets are lost, in which case the j packet losses are irrecoverable. Eq. (4) can be simplified yielding

$$\mathbb{L}_e = \mathbb{L}_c \left[1 - (1-\mathbb{L}_c)^k\right] \tag{5}$$

Eq. (5) states that a data packet is irrecoverably lost only if it is lost in the core (with probability $\mathbb{L}_c$) **and** not all other k packets in the window (this includes the FEC packet) are successfully received (otherwise the lost data packet can be reconstructed).

Eq. (5), in conjunction with Eq. (3) and Eq. (2), can be used to directly estimate the edge-to-edge loss $\mathbb{L}_e$ as a function of FEC block-size k. In Fig. 7 we plot on log-scale the edge-to-edge packet loss probability as a function of the block-size k for different values of the offered load ρ (10% to 40%). An important observation to emerge from this plot is that for a given load, the loss decreases with block-size k, reaches a minimum, and then starts increasing as the block-size gets larger. This provides an analytical explanation of why the simulation plot in Fig. 5 shows TCP goodput to first increase and then fall with block-size k, as TCP throughput is inversely related to the square root of end-to-end packet loss

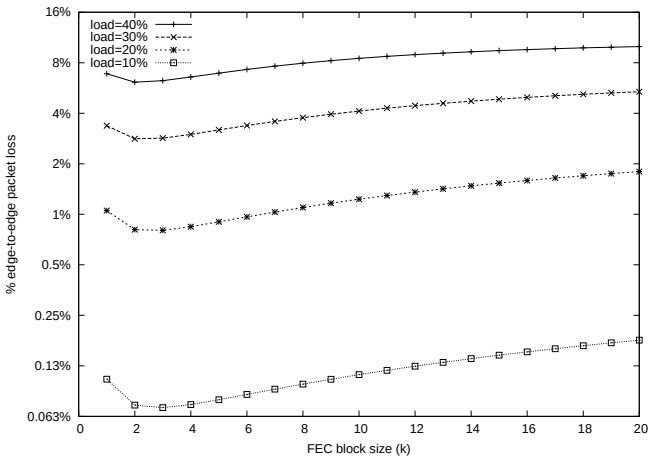

Fig. 7. % edge-to-edge packet loss for different offered loads from analysis

[24]. The figure also gives us some estimate of the strength of FEC required ($k = 2, 3$ depending on the offered load) for minimising loss: the recovery benefit of stronger FEC (i.e. lower k) is outweighed by the overhead it introduces in terms of load, while weaker FEC (i.e. larger k) does not sufficiently recover lost data packets.

C. Unfairness in a Multi-Hop Network

Having seen the benefits offered by FEC for a single link, we now evaluate its performance on a more general wide-area network topology. To this end, we choose the NSFNet topology shown in Fig. 2 as our representative core network, which is made up of core routers (numbered 0 to 13) and single buffer links interconnecting them. The numbers along the core links indicate the propagation delay in milliseconds. We consider ten edge links feeding traffic into every core router, and each edge router in turn is fed by five access links (for clarity the figure shows a smaller number of access/edge links). All core links operate at 1 Gbps, all edge links at 100 Mbps, and the access link rates are uniformly distributed between $[7, 10]$ Mbps, to reflect a typical home user. These numbers ensure that the core is not the bottleneck for any TCP flow. The destination end-hosts are chosen randomly such that every flow traverses at least one hop on the core network; in all there are 3480 TCP flows in the network comprising of 784 one-hop flows, 1376 two-hop flows and 1320 three-hop flows. We assume all flows to be long-lived (Section V describes results when both short-lived and long-lived TCP flows coexist). Data in the interval $[20, 35]$ sec is used for the computations to capture the network's steady state.

Fig. 8 plots the ratio of average goodput with FEC to the average goodput with delay-bandwidth buffers for 1-, 2- and 3-hop TCP flows as a function of block-size - goodput with delay-bandwidth buffers is used as the benchmark because core routers today have large buffers [25]. Core link loads were found to vary between 7% and 38% with large buffers (the average being $\approx 24\%$). These numbers fall in the regime in which most ISPs operate their networks today. The figure also depicts, via horizontal lines, the goodput ratios in the non-FEC case.

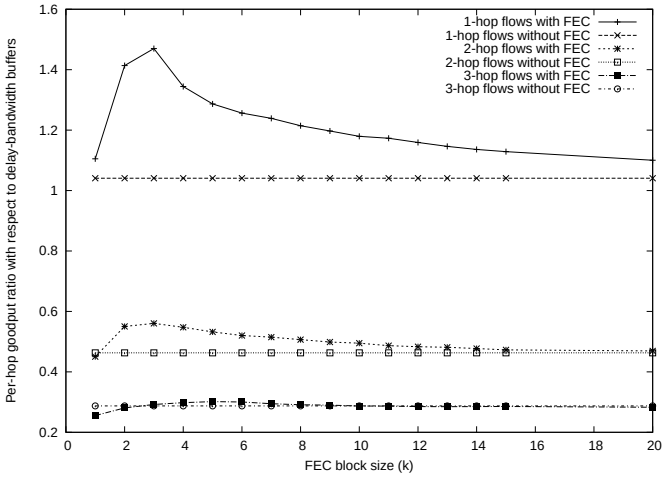

Fig. 8. Ratio (average goodput with FEC to average goodput with delay-bandwidth buffers) for 1-, 2-, 3-hop TCP flows on NSFNet topology

Fig. 8 shows that on average, goodput for 1-hop flows with FEC (at $k = 3$) is nearly 1.5 times that of delay-bandwidth buffers, meaning that FEC enabled 1-hop flows perform better in a bufferless network than in a fully-buffered network. The ratio reduces to 0.56 for 2-hop flows and to 0.3 for 3-hop flows, indicating that TCP performance degrades rapidly with hop-length.

To see if this large degree of unfairness is predominantly due to the closed-loop nature of TCP traffic or if the same phenomenon can be observed with open-loop UDP traffic, we simulate Poisson flows with a mean rate of 1 Mbps using the same simulation setting as before. Our observation is that 1-, 2- and 3-hop Poisson flows, without FEC, achieve goodput ratios of 0.98, 0.97 and 0.95 respectively, and with FEC these numbers reach above 0.98 for all flows, indicating that unlike TCP flows, the fairness for UDP flows is not adversely affected by hop-length.

To explain why multi-hop TCP flows perform so poorly, we plot in Fig. 9 the histogram of edge-to-edge packet loss (for the non-FEC case) for flows with different hop-lengths. We can observe that while over 95% of 1-hop flows experience loss

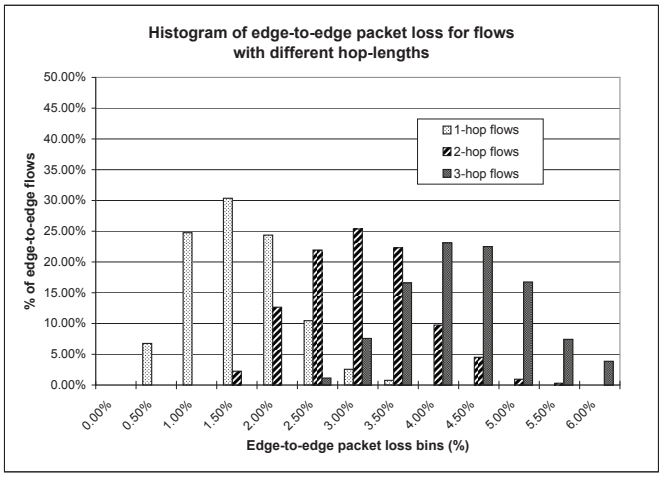

Fig. 9. Histogram of edge-to-edge packet loss (from simulation) for TCP flows with different hop lengths (and no FEC) on the NSFNet

only in the range 0.5-3%, it increases to 1.5-4.5% for 2-hop flows, and further to 2.5-6% for 3-hop flows. To appreciate the impact these numbers have on the edge-to-edge performance, we note that a doubling in the loss rate from 1% to 2% reduces the throughput of open-loop UDP traffic by only 1%, whereas the throughput for closed-loop TCP traffic drops by 30% (and goodput by even more as seen in Fig. 8), since the average throughput of a TCP flow in the congestion avoidance mode is inversely proportional to the square root of packet loss. The relatively higher loss rates for 2- and 3-hop flows result in their fair-share of the bandwidth being unfairly utilised by 1-hop flows (since TCP is inherently greedy and is designed to exploit as much of the bandwidth as available), leading to unfairness. These observations motivate us to devise a scheme that provides fairness to flows of different hop-lengths, as described in the next section.

V. Tuning FEC for Network-Wide Fairness

We observed from the results in the previous section that in a bufferless network, multi-hop TCP flows can experience significantly lower end-to-end goodput than single-hop flows, leading to unfairness. In this section, we address this deficiency by developing a framework that ensures fairness to both single- and multi-hop flows. It is important to note that determining the optimal block-size settings to ensure fairness for the various flows in the network is non-trivial because for any edge-to-edge flow, its optimal FEC strength not only depends on its offered load, but also on the loads that the flow sees along the links in its routing path, which in turn depends on the offered load and the FEC strength used by the other flows traversing those links. We now develop a global optimisation framework and propose a practical algorithm that determines FEC strengths needed to achieve fairness amongst the different single-/multi-hop flows in a network.

A. A Global Optimisation Framework

Let $\lambda_{i,j}$ denote the offered load (without FEC) to the core network by the edge-to-edge flow between ingress router i and egress router j, henceforth represented as (i,j). Let $\overline{\lambda_{i,j}}$ be the load to the core network when the flow employs FEC using $k_{i,j}$ as its block-size. Consequently,

$$\overline{\lambda_{i,j}} = \left(\frac{k_{i,j}+1}{k_{i,j}}\right)\lambda_{i,j} \tag{6}$$

Using Eq. (3), we can compute the packet loss probability $L_c^{u,v}$ at a core link (u,v) as

$$L_c^{u,v} = \frac{\left(\sum_{(u,v)\in r(i,j)} \overline{\lambda_{i,j}}\right)^2}{1+\left(\sum_{(u,v)\in r(i,j)} \overline{\lambda_{i,j}}\right)+\left(\sum_{(u,v)\in r(i,j)} \overline{\lambda_{i,j}}\right)^2} \tag{7}$$

where $r(i,j)$ is the routing path of edge-to-edge flow (i,j). In general, a flow can traverse multiple hops on the core network before reaching the egress edge router, and we assume that edge-to-edge losses are independent and additive over the links the flow traverses (because core loss in practice will be reasonably small). Therefore, denoting $\mathbb{L}_c^{i,j}$ to be the aggregate core path loss probability for the flow (i,j),

$$\mathbb{L}_c^{i,j} = \sum_{(u,v)\in r(i,j)} L_c^{u,v} \tag{8}$$

We can now compute $\mathbb{L}_e^{i,j}$, the edge-to-edge packet loss probability for flow (i,j) by substituting in Eq. (5) the core path loss probability for the flow derived from Eq. (8). Thus,

$$\mathbb{L}_e^{i,j} = \mathbb{L}_c^{i,j}\left[1-\left(1-\mathbb{L}_c^{i,j}\right)^{k_{i,j}}\right] \tag{9}$$

We now capture the notion of fairness by formulating an optimisation problem as follows.

Inputs:

- Offered load $\lambda_{i,j}$ by every edge-to-edge flow (i,j).
- $r(i,j)$ the routing path of the flow (i,j).

Objective function:

$$\min_{k_{i,j}} \left(\max_{i,j} \mathbb{L}_e^{i,j}\right) \tag{10}$$

Subject to: $k_{i,j} \in \{1,2,3,\ldots\}$

Output: Set $\{k_{i,j}\}$, which denotes the optimum FEC block-size for every edge-to-edge flow.

The optimisation objective in Eq. (10) attempts to choose the set of FEC strengths $k_{i,j}$ that achieves min-max loss fairness, i.e. minimises the maximum loss rate over all flows, irrespective of hop-length. We do note that other definitions of fairness exist in the literature, such as proportional fairness [26]. Study of FEC settings for generalised frameworks of fairness based on utility functions [27] are beyond the scope of the current study and are left for future work.

The optimisation formulation above has a non-linear objective function, since the block-sizes $k_{i,j}$ are in the exponent of Eq. (9), and additionally has integrality constraints on the $k_{i,j}$'s, rendering the problem intractable for large networks. In what follows we propose a practical heuristic that bins flows according to their edge-to-edge loss rate and assigns identical FEC strength to flows in the same bin.

Algorithm 1 Loss-rate based heuristic

Inputs: Offered load $\lambda_{i,j}$ for every edge-to-edge flow (i,j),
Routing path $r(i,j)$ of flow (i,j),
Bin granularity p
Output: The FEC block-size for all the edge-to-edge flows.
Block-size set $K = \{1,2,3,4,5,6,8,10,15,20,25,100\}$
minLoss $\leftarrow$ 1, optimalBlockSizes $= \phi$

1: **for** every link (u,v) in the core network **do**
2: use $\lambda_{i,j}$ (instead of $\overline{\lambda_{i,j}}$) and estimate the loss rate at link (u,v) using the loss expression in Eq. (7).
3: **end for**
4: **for** every edge-to-edge flow (i,j) **do**
5: determine its total end-to-end loss using Eq. (8).
6: **end for**
7: create n bins $1,2,\ldots,n$, where $n = \lceil \text{MaxLoss}/p \rceil$; bin i corresponds to loss rate in the range $[p(n-i), p(n+1-i)]$ (note that bins are arranged in descending order of loss). Assign each flow to its respective loss bin.
8: let $k_1, k_2, \ldots, k_n$ be the block-sizes for all flows in bins $1,2,\ldots,n$, respectively.
9: **for** $k_1 \in K$ **do**
10: **for** $k_2 \geq k_1$ and $k_2 \in K$ **do**
11: **for** $k_3 \geq k_2$ and $k_3 \in K$ **do**
12: $\vdots$
13: **for** $k_n \geq k_{n-1}$ and $k_n \in K$ **do**
14: **for** every edge-to-edge flow (i,j) **do**
15: using Eq. (7)-(9) compute its net loss rate, where $k_{i,j}$ is one of $k_1, k_2, \ldots, k_n$, depending on which loss bin the flow belongs to.
16: **end for**
17: maxLoss $\leftarrow$ maximum loss across all flows
18: **if** maxLoss $<$ minLoss **then**
19: minLoss $\leftarrow$ maxLoss
20: optimalBlockSizes $\leftarrow [k_1, k_2, \ldots, k_n]$
21: **end if**
22: **end for**
23: **end for**
24: **end for**
25: **end for**
26: output optimalBlockSizes

B. Loss-rate Based Heuristic

As mentioned earlier, the best block-size to use for an edge-to-edge flow depends on the load on each of the links it goes through (and hence the loss rate that the flow experiences), which in turn depends on the offered load and the FEC strength of the other flows traversing those links. We now outline a practical heuristic algorithm that operates on the rationale that flows experiencing similar edge-to-edge loss should be protected using similar FEC strengths. Consequently, we assign (the potentially large number of) flows to a small number of bins and determine optimal FEC strength assignments to bins that achieve min-max fairness. The pseudo code for the proposed heuristic algorithm is shown in Algorithm 1. The offered load between edge-to-edge pairs and the routing path of flows on the core network are known aforehand, which allows us to estimate the load on each core link by summing the offered loads of all the flows that traverse the link. Steps 1-3 of our algorithm compute the loss rate on each core link using Eq. (7), where $\overline{\lambda_{i,j}}$ can be replaced with the $\lambda_{i,j}$, i.e. the offered load of the edge-to-edge flow (without FEC). Steps 4-6 compute the edge-to-edge loss rate for each flow by summing the loss rates along the links on its routing path (under the assumption that the losses on each link are independent and small). Step 7 assigns flows to bins based on their estimated end-to-end loss rates. The number of bins is determined based on the chosen bin width (input to the algorithm) and the maximum flow loss *MaxLoss* in the network. For example, for the NSFNet topology we consider in this paper, maximum loss rate (observed in simulation and confirmed by analysis) was around 6%, and we therefore consider twelve bins, each of width 0.5%. In steps 9-25, the algorithm tries all combinations of FEC strengths assigned to each bin to identify the one that minimises the maximum flow loss. Assigning the same FEC strength to all flows with similar edge-to-edge loss (i.e. in the same bin) dramatically reduces the search space of FEC block-size assignments, while still achieving fairness by allowing flows with different loss performance to have different FEC strengths. Steps 9-13 show that the search space can be further reduced by making the reasonable assumption that flows with lower loss (higher numbered bin) will require weaker FEC (larger block-size) than flows with higher loss (in lower numbered bins). Steps 14-16 recompute edge-to-edge loss for each flow for the global choice of FEC strengths,

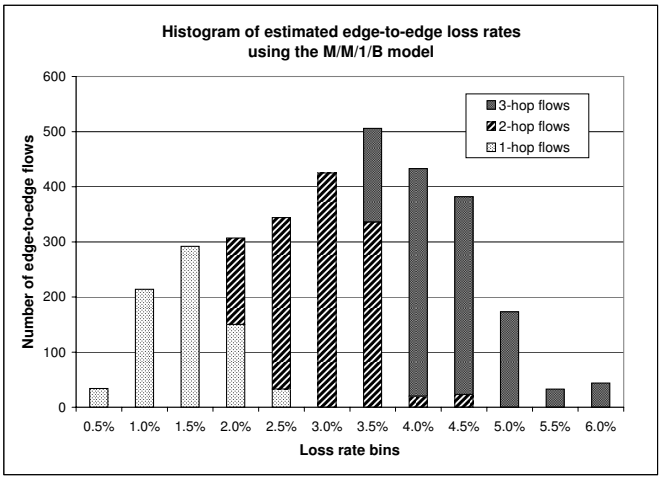

Fig. 10. Histogram of estimated edge-to-edge loss using the M/M/1/B model

Network setting	Average goodput (Mbps)			Fairness Index
	1-hop flows	2-hop flows	3-hop flows	
No FEC	1.571	0.667	0.391	0.65
FEC block-size 3 for all flows	2.219	0.807	0.397	0.58
FEC block-sizes from our heuristic	1.348	0.678	0.613	0.73
delay-bandwidth buffers	1.509	1.440	1.359	1

Fig. 11. Average goodputs and fairness indices for long-lived TCP flows

and steps 17-20 store the best choice seen so far. The set of optimal block-sizes for each bin that minimises the maximum edge-to-edge loss is output in step 26.

We now apply the above heuristic to the NSFNet network topology. The load offered by each edge-to-edge flow is dictated by the network topology and TCP dynamics, and in our case is derived (as described in Sec. IV-C) from measurement in simulation of the single buffer core network (much the same way as ISP would know loads by measurement in their network). Using this offered load, the estimated loss rates for the edge-to-edge flows are computed using Eq. (7) and (8). These losses are shown as a histogram in Fig. 10, comprising 12 loss bins of width 0.5% each, covering loss rates in the range 0-6%. The histogram also shows the composition of each bin in terms of the flow hop-lengths, and indicates that flows with larger hop-length experience higher loss, in accordance with the simulation measurements shown in Fig. 9. In general we found that the end-to-end losses estimated from our analysis corroborated well with simulation, despite the simplifying assumptions that ignore TCP dynamics.

We applied our heuristic to search for optimal block-sizes over the 12 loss bins. If the number of loss bins is n and the number of elements in the block-size set K is $|K|$, then the worst case time complexity of our heuristic is $O(|K|^n)$. On a PC with a 3 GHz AMD Athlon processor and 4 GB RAM, the search took around 4 hours, and gave us optimal block-size settings of $\{2, 2, 2, 3, 5, 5, 8, 10, 25, 25, 25, 25\}$, meaning, use block-size 2 for all flows that experience loss rates in the range 4.5-6%, 3 for flows with loss between 4-4.5%, 5, 8 and 10 for flows with loss 3-4%, 2.5-3%, and 2-2.5%, and 25 for the remaining flows.

C. Simulation Results and Fairness for the NSFNet Network

We incorporated the above FEC strengths output by our heuristic into our simulations of TCP traffic on the NSFNet topology. To evaluate if our framework for minimising the maximum edge-to-edge loss is effective in achieving fairness in goodput for TCP flows, we employ the widely used Jain's fairness index [28] as an indicator of the heuristic's performance. The fairness index is a real number between 0 and 1, with a higher value indicating better fairness. Our benchmark for comparison is the goodput of TCP flows when core links have large (delay-bandwidth) buffers.

Fig. 11 shows the fairness index for four pertinent scenarios - no FEC, FEC with $k = 3$ for all flows, FEC with our heuristic, and large buffers. The following three observations emerge:

1) First, in a network without FEC, the fairness index is low at 0.65, and 1-hop flows get higher goodput when the network has near-zero buffers than in a fully-buffered network. This comes at the expense of greatly reduced goodput for multi-hop flows, with 3-hop flows getting only about 29% of the goodput they would compared to a network with large buffers. This highlights the unfairness amongst flows based on their hop-lengths.

2) When the FEC block-size is set uniformly at $k = 3$ across all flows, performance improves for all flows, but the unfairness is exacerbated due to 1-hop flows reaping most of the benefits.

3) Finally, setting the block-sizes according to our heuristic results in a higher fairness index of 0.73. The algorithm is instrumental in restraining 1-hop flows while helping 3-hop flows reach up to 45% of their ideal value; outperforming the two previous scenarios by more than 50%.

These results demonstrate that careful configuration of FEC strengths is key to realising fair performance in future buffer-less core networks. To understand the sensitivity of our results to the exact choice of block-sizes, we studied the impact a slight perturbation to the block-size settings has on the fairness index. We ran ten simulations, each time varying the FEC strength of several bins by a small amount. We found the index to vary between 0.70 and 0.76 (i.e. $\pm 4\%$ of 0.73 from our heuristic), suggesting that the block-sizes output by our algorithm are stable.

D. Mix of Short-Lived and Long-Lived TCP Flows

Our study of FEC has thus far only considered long-lived TCP flows, and we now consider realistic mixes comprising of long- and short-lived TCP flows, wherein the number of active TCP flows is time-varying. Measurement studies in the Internet core show that a large number of TCP flows (e.g., HTTP requests) are short-lived and carry only a small

Network setting	Average goodput (Mbps)			Fairness Index
	1-hop flows	2-hop flows	3-hop flows	
No FEC	1.264	0.683	0.440	0.76
FEC block-sizes from our heuristic	1.109	0.651	0.532	0.78
delay - bandwidth buffers	0.992	0.866	0.729	1

Fig. 12. Average goodputs and fairness indices for short-lived TCP flows

volume of traffic, while a small number of TCP flows (e.g., FTP) are long-lived and carry a large volume of traffic. To incorporate such realistic TCP traffic we simulate the closed-loop flow arrival model described in [29], operating as follows. A given number of users perform successive file transfers to their respective destination nodes. The size of the file to be transferred follows a Pareto distribution with mean 100 kB and shape parameter 1.2. These chosen values are representative of Internet traffic, and comparable with measurement data. After each file transfer, the user transitions into an idle ("thinking period") or off state. The duration of the "thinking period" is exponentially distributed with mean 1 sec. We implemented this model in *ns-2* and repeated our simulations of the NSFNet topology as described in Section IV-C, setting 80% (2761 out of 3480) of the TCP flows to be short-lived, with the remaining 719 being long-lived.

Our heuristic for selecting FEC strengths is oblivious to the nature of the traffic, and hence operates as before, using as input the modified load conditions arising from the short-lived TCP flows. Flow losses estimated from our M/M/1/B model are in the range of 0-3.5%, which matches well with the flow losses measured in simulation. Our heuristic uses $n = 7$ loss bins in increments of 0.5%, and yields optimal block-size settings of $\{2, 3, 4, 6, 15, 25, 100\}$. In other words, flows with the highest loss rate should use strong FEC with block-size 2, while flows experiencing low loss use very weak FEC (block-size of 100).

Fig. 12 shows the relative goodputs and fairness indices for 2761 short-lived flows under two near-zero buffer scenarios of no FEC, and FEC strengths from our heuristic, as well as the benchmark performance for a fully-buffered network. We make three observations from it:

1) In a network without FEC, goodput for 1-hop flows is 27% higher than in a network with large buffers. This benefit comes at the expense of multi-hop flows; average goodput of 3-hop flows is $\approx 40\%$ lower than the benchmark goodput. The fairness index is 0.76.

2) We observe that our heuristic assignment of FEC strengths achieves a better fairness index than not having FEC. More importantly, the average goodputs of 1-, 2-, and 3-hop flows improve considerably, coming to within 12%, 25%, and 27% of their respective benchmark goodputs (i.e. with large buffers). We believe this is a significant achievement given that buffers are eliminated from the core network. 3-hop flows with FEC outperform the no FEC case by over 20%.

3) We note that the improvement in fairness index (over no FEC) when including for short-lived flows (Fig. 12) is not as dramatic as in our earlier scenario that considered only long-lived TCP flows (Fig. 11). This is because our optimisation does not account for the dynamic nature of traffic loads. Reconfiguring FEC strengths on-the-fly to adapt to dynamic loading conditions has the potential to improve performance considerably; we will examine it as part of future work.

Our results thus far used fairness index as the main indicator of performance. Another metric, average flow completion time (AFCT), namely the time it takes to transfer files, is also believed to be a good measure of performance for short flows [21]. Our simulations showed that 3-hop flows had an AFCT of 0.99 sec when the network has large buffers. With near-zero buffers, AFCT rises by nearly 50% (to 1.48 sec) without FEC, and by 57% (to 1.55 sec) when all flows use block-size 3. FEC strengths from our heuristic reduces AFCT to 1.24 sec (only 25% higher than with large buffers), which is substantially better than having no FEC or using uniform FEC.

For 2-hop flows, the AFCT with large buffers is 0.86 sec, and this increases by 19% (to 1.03 sec) when using FEC strengths from our heuristic. Finally, 1-hop flows have a slightly reduced AFCT (0.65 sec) in a near-zero buffer network than in a fully-buffered network (0.75 sec) as they capitalise on the capacity left by longer hop flows in such a network. Overall the AFCT numbers, much like the goodput values, are encouraging, and illustrate that edge-to-edge FEC when configured well can be effective in realising acceptable TCP performance for all flows in a bufferless core network.

VI. Conclusions and Future Work

Optical switching is expected to play a major role in scaling the capacity of future core routers. However, buffering packets in the optical domain is a very challenging operation. In this paper, we envisioned a bufferless core optical network and made three new contributions: (1) we proposed a novel edge-to-edge packet-level FEC mechanism as a means of battling high core losses, (2) we considered a realistic core network (NSFNet), developed an optimisation framework, and designed a practical heuristic algorithm to improve fairness between single- and multi-hop flows, and (3) we studied the performance of FEC for realistic mixes of short- and long-lived TCP flows and showed that the FEC scheme can be tuned to yield good performance.

This paper is a first step towards understanding the impact of packet-level FEC in a bufferless core network. Future work includes obtaining accurate estimates of the offered load by modelling the feedback nature of TCP. We have considered the FEC block-sizes to be static; one could extend the FEC scheme to incorporate dynamic adaptation of the FEC strength (block-size) based on on-the-fly measurement of actual losses in the network. The benefits of sophisticated schemes such as Reed-Solomon codes along with experimentation are also valuable research directions.

REFERENCES

[1] Cisco white paper, *Approaching the zettabyte era*, http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481374.pdf, Jun 2008.

[2] G. Appenzeller, I. Keslassy and N. McKeown, "Sizing router buffers," *Proc. ACM SIGCOMM*, USA, Aug-Sep 2004.

[3] J. Chabarek et al., "Power awareness in network design and routing," *Proc. IEEE INFOCOM*, USA, Apr 2008.

[4] A. Vishwanath et al., "Adapting router buffers for energy efficency," *Proc. ACM SIGCOMM CoNEXT*, Japan, Dec 2011.

[5] I. Keslassy, S. Chuang, K. Yu et al., "Scaling internet routers using optics," *Proc. ACM SIGCOMM*, Germany, Aug 2003.

[6] P. Bernasconi et al., "Architecture of an integrated router interconnected spectrally (IRIS)," *Proc. IEEE HPSR*, Poland, Jun 2006.

[7] D. J. Blumenthal, "Optical labeled packet switching and the LASOR project," *Proc. 17th Annual Meeting of the IEEE Lasers and Electro-Optics Society (LEOS)*, Nov 2004.

[8] E. F. Burmeister et al., "SOA gate array recirculating buffer for optical packet switching," *Proc. IEEE/OSA OFC*, USA, Feb 2008.

[9] J. D. LeGrange et al., "Demonstration of an integrated buffer for an all-optical packet router," *IEEE Photonic Technology Letters*, vol. 21, no. 2, pp. 781-783, Jun 2009.

[10] *CAIDA packet length distributions*, Available online: http://www.caida.org/research/traffic-analysis/AIX/plen_hist/

[11] A. Vishwanath, V. Sivaraman and M. Thottan, "Perspectives on router buffer sizing: Recent results and open problems," *ACM SIGCOMM CCR Editorial Zone* vol. 39, no. 2, pp. 34-39, Apr 2009.

[12] E. M. Wong et al., "Towards a bufferless optical internet," *IEEE/OSA Journal of Lightwave Technology*, vol. 27, no. 4, pp. 2817-2833, Jul 2009.

[13] V. Sivaraman et al., "Packet pacing in small buffer optical packet switched networks," *IEEE/ACM Transactions on Networking*, vol. 17, no. 4, pp. 1066-1079, Aug 2009.

[14] O. Tickoo, V. Subramanian, S. Kalyanaraman, and K. K. Ramakrishnan, "LT-TCP: End-to-end framework to improve TCP performance over networks with lossy channels," *Proc. IEEE IWQoS*, Germany, Jun 2005.

[15] L. Kontothanassis et al., "A transport layer for live streaming in a content delivery network", *Proceedings of the IEEE*, vol. 92, no. 9, pp. 1408-1419, Sep 2004.

[16] R. S. Prasad, C. Dovrolis and M. Thottan, "Router buffer sizing for TCP traffic and the role of the output/input capacity ratio," *IEEE/ACM Transactions on Networking*, vol. 17, no. 5, pp. 1645-1658, Oct 2009.

[17] M. E. Engelhardt, "Events in time: Basic analysis of Poisson data," *Idaho National Engineering Laboratory*, Avilable online: http://www.osti.gov/bridge/servlets/purl/10191309-1p5pjN/webviewable/10191309.pdf, 1994.

[18] A. Vishwanath, V. Sivaraman and M. Thottan, "On the energy-efficiency of a packet-level FEC based bufferless core optical network," *Proc. IEEE/OSA Optical Fiber Communications (OFC) Conference*, USA, Mar 2012.

[19] A. Odlyzko, "Data networks are lightly utilized, and will stay that way," *Review of Network Economics*, vol. 2, no. 3, pp. 210-237, Sep 2003.

[20] N. Dukkipati and N. McKeown, "Why flow-completion time is the right metric for congestion control", *ACM SIGCOMM CCR*, vol. 36, no. 1, pp. 59-62, 2006.

[21] A. Lakshmikantha, R. Srikant and C. Beck, "Impact of file arrivals and departures on buffer sizing in core routers" *Proc. IEEE INFOCOM*, USA, Apr 2008.

[22] A. Vishwanath, V. Sivaraman and D. Ostry, "How Poisson is TCP traffic at short time-scales in a small buffer core network?" *Proc. IEEE Advanced Networks and Telecommunication Systems (ANTS)*, India, Dec 2009.

[23] L. Kleinrock, "Queueing Systems. Volume 1: Theory," *Wiley-Interscience publication*, 1975.

[24] M. Mathis, J. Semke, J. Madhavi, and T. Ott, "The macroscopic behavior of the TCP congestion avoidance algorithm," *ACM SIGCOMM CCR*, vol. 27, no. 3, pp. 67-82, 1997.

[25] C. Villamizar and C. Song, "High performance TCP in ANSNet," *ACM SIGCOMM CCR*, vol. 24, no. 5, pp. 45-60, 1994.

[26] F. Kelly, A. Maulloo and D. Tan, "Rate control for communication networks: Shadow prices, proportional fairness and stability," *Journal of Operations Research Society*, vol. 49, pp. 237-252, 1998.

[27] R. Srikant, "The mathematics of Internet congestion control," *Birkhäuser*, 2004.

[28] R. Jain, D. Chiu and W. Hawe, "A quantitative measure of fairness and discrimination for resource allocation in shared computer system," *DEC technical report* TR-301, 1984.

[29] B. Schroeder, A. Wierman, and M. Harchol-Balter, "Closed versus open: A cautionary tale," *Proc. USENIX NSDI*, USA, May 2006.

[30] A. Vishwanath, V. Sivaraman, M. Thottan and C. Dovrolis, "Enabling a bufferless core network using edge-to-edge packet-level FEC," *Proc. IEEE INFOCOM*, USA, Mar 2010.

Arun Vishwanath (M '11) is a Senior Research Fellow at the Centre for Energy-Efficient Telecommunications, University of Melbourne. He received the Ph.D. degree in Electrical Engineering from the University of New South Wales in Sydney, Australia, in 2011. He was a visiting Ph.D. scholar in the Department of Computer Science at North Carolina State University, USA in 2008. His research interests include energy-efficient networking, network design, optical networks and router buffer sizing.

Vijay Sivaraman (M '94) received his B. Tech. degree from the Indian Institute of Technology in Delhi, India, in 1994, his M.S. from North Carolina State University in 1996, and his Ph.D. from the University of California at Los Angeles in 2000, all in Computer Science. He has worked at Bell-Labs and a silicon valley startup manufacturing optical switch-routers. He is now an Associate Professor at the University of New South Wales in Sydney, Australia. His research interests include optical networking, packet switching and QoS routing.

Marina Thottan (M '00) received the Ph.D. degree in electrical and computer systems engineering from Rensselaer Polytechnic Institute in Troy, NY in, 2000. She is Director of the Mission-Critical Communications and Networking Group, Bell Laboratories, Murray Hill, NJ. Most recently, she has been leading work on smart grid communication networks. She has published over 40 papers in scientific journals, book chapters, and refereed conferences. Dr. Thottan is a Member of the Association for Computing Machinery.

Constantine Dovrolis (M '01) is an Associate Professor at the College of Computing of the Georgia Institute of Technology. He received the Computer Engineering degree from the Technical University of Crete in 1995, the M.S. degree from the University of Rochester in 1996, and the Ph.D. degree from the University of Wisconsin-Madison in 2000. He joined Georgia Tech in August 2002, after serving at the faculty of the University of Delaware for about two years. He has held visiting positions at Thomson Research in Paris, Simula Research in Oslo, and FORTH in Crete. His current research focuses on the evolution of the Internet, Internet economics, and on applications of network measurement. He is also interested in network science and in applications of that emerging discipline in the understanding of complex systems.