On the Energy-Efficiency of a Packet-Level FEC Based Bufferless Core Optical Network

Arun Vishwanath*, Vijay Sivaraman^, and Marina Thottan#
** Centre for Energy-Efficient Telecommunications, Dept. of Electrical and Electronic Engineering, University of Melbourne, Australia*
^ School of Electrical Engineering and Telecommunications, University of New South Wales, Sydney, Australia
# Bell Labs Alcatel-Lucent, Murray Hill, NJ, USA
arun.v@unimelb.edu.au, vijay@unsw.edu.au, marina.thottan@alcatel-lucent.com

Abstract: We proposed packet-level forward error correction (FEC) as a promising technique for enabling bufferless core optical networks. In this paper, we systematically analyse its energy-efficiency and show that it can produce substantial power savings.
OCIS codes: (060.4259) Networks, packet-switched; (060.1810) Buffers, couplers, routers, switches, and multiplexers

1. Introduction

Improving the energy-efficiency of core routers has attracted widespread attention over the past few years. While various approaches have been proposed to tackle this problem, our focus is on router packet buffers, which are of the order of a few Gigabytes, and accounts for nearly 10% of the power consumed by a line-card [1]. It is well-known that core links are typically over-provisioned, and ISPs often operate their core networks at only 20-30% utilisation. Under such circumstances, large buffers from ***core routers*** can virtually be eliminated (thereby improving the overall energy-efficiency) without adversely impacting traffic performance [2]. Our mechanism to do so relies on a novel edge-to-edge based packet-level FEC scheme, described next.

2. The Edge-to-Edge Packet-Level FEC Framework

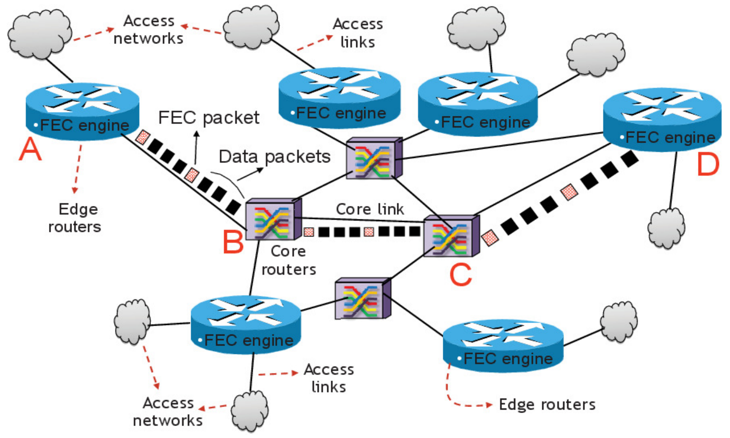

Fig. 1: Topology to illustrate our edge-to-edge packet-level FEC framework

Fig. 1 shows a small segment of an ISP's network comprising of access networks (e.g. home users), electronic edge routers and bufferless optical core routers. FEC is performed on the ***aggregate*** traffic flowing from an ingress to an egress edge router. The FEC packet is computed by bit-wise XOR-ing a block of k successive data packets (k denotes the FEC block-size). The ingress edge therefore transmits one FEC packet for every block of k data packets, permitting the corresponding egress edge to recover from at most one lost data packet from the block. To illustrate our scheme, the figure shows traffic flowing from ingress edge router A to egress edge router D along the path A-B-C-D. Assuming the block-size is three, A keeps a running XOR of the 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 the packets it receives from A. If it receives all but one data packet in the window of four packets (known via a block identifier, blockID, 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. We showed in [2] that this XOR-based scheme can be tuned to deliver good end-to-end TCP performance; over 90% of the Internet traffic today is TCP.

3. Steps Involved During FEC

We first examine the ingress FEC operations (in 3.1), and explore the egress functionality (in 3.2), respectively.

3.1. FEC at Ingress Edge Router A

The ingress router A first receives a data packet from an access network. The network processor then performs an IP lookup to determine the next-hop interface, which also returns the ID of the egress edge router (D in this e.g.). The FEC block-sizes for the various edge-to-edge flows are preconfigured in all edge routers according to [2], and the FEC engine at the ingress keeps track of the number of the XORs performed (in register variable numXORs). Then:

- *Step 1:* FEC packet (from SRAM) and the data packet (from DRAM) are *read* and bit-wise *XOR*-ed. *Increment* numXORs by one. W*rite* blockID into the data packet header.
- *Step 2:* C*ompare* numXORs with the block-size *k* (in this e.g. *k*=3).
- *Step 3:* IF true, THEN
  (a) *Read* from SRAM and TCAM respectively the egress router's IP address and next-hop MAC address, and *write* into SRAM to create the FEC packet's IP and MAC headers, (b) Update FEC header to include the blockID (subsequently *incremented* by one to mark the start of the next block), and (c) T*ransmit* FEC packet and clear corresponding SRAM location (*write* zeroes).
- ELSE
- *Step 4:* The XOR-ed FEC packet is *written* back into SRAM.

3.2. FEC at Egress Edge Router D

Packets from A to D can be dropped in the core (i.e. on links A-B, B-C, or C-D) because these links are devoid of any buffering capability. The FEC engine at the egress keeps track of the expected block ID (in register variable expBlockID) and the number of data packets lost within a block (in register variable numDataPktsLost). Then:

- *Step 1:* IF an FEC packet is received but (its blockID $\neq$ expBlockID or numDataPktsLost $\neq$ 1), THEN
  (a) Clear the running XOR packet (*write* zeroes in the corresponding SRAM location), (b) expBlockID $\leftarrow$ *add* (blockID,1), and (c) numDataPktsLost $\leftarrow$ *k*.
  ELSE
- *Step 2:* (a) Running XOR packet (from SRAM) and the incoming FEC packet (from DRAM) are *read* and bit-wise *XOR*-ed to recover the lost data packet, (b) running XOR is cleared (*write* zeroes in the SRAM location), (c) numDataPktsLost $\leftarrow$ *k*, and (d) *increment* expBlockID.
- *Step 3:* IF a data packet is received and (its blockID == expBlockID), THEN
  (a) Running XOR packet (from SRAM) and the data packet (from DRAM) are *read* and XOR-ed bit-wise, (b) *write* the XOR packet back into SRAM, and (c) *subtract* (numDataPktsLost,1).
  ELSE */* new data block starts, clear running XOR */*
- *Step 4:* (a) Running XOR is *read* from SRAM and cleared, (b) it is then *XOR*-ed with the data packet and the result *written* into SRAM, (c) expBlockID $\leftarrow$ blockID, and (d) numDataPktsLost $\leftarrow$ (*k-1).*

4. Estimating the Power Consumption of the FEC Framework

As the FEC operations are performed by the router's hardware, estimating the power consumed by the FEC logic at the ingress/egress necessitates analysing the energy associated with native hardware operations such as reading/writing a bit from/to memory (consisting of memory controllers and SRAM/DRAM chips), bit-level XORs, comparators/increment operations, as well as transmitting/receiving the FEC packet (i.e. operations in *Steps 1-4*).

Denote by $E_I(N)$ and $E_E(N)$ the respective energy costs/packet at the ingress and egress when step $N \in \{1,2,3,4\}$ is executed (in Sections 3.1 and 3.2 above). Then, for given block-size *k* and data rate *f* (in packets/sec),

At the ingress: *Steps 1, 2,* are executed for every incoming data packet, *Step 3* is executed once per *k* data packets, and *Step 4,* the rest of the time. Therefore, the total power consumption at the ingress, P_I, can be expressed as:

$$P_I = f\{E_I(1) + E_I(2) + E_I(3)\cdot 1/k + E_I(4)\cdot (k-1)/k\} \quad (1)$$

At the egress: Let *p* denote the core packet loss rate. *Steps 1, 2* are executed when a FEC packet arrives, which happens with probability 1/(*k*+1). Further, the probability that its blockID is not equal to expBlockID is p^{k+1} since this happens only when the expected FEC packet and all subsequent data packets (from the same block) are lost (the probability of two or more contiguous block losses is negligible). Finally, the probability that exactly one data packet is missing from a block of *k* packets is $\binom{k}{1}p(1-p)^{k-1}$. Combining these probabilities, we can express the power consumed by *Steps 1, 2* as:

$$f\left\{\frac{1}{k+1}\left[\left\{p^{k+1}+1-\binom{k}{1}p(1-p)^{k-1}\right\}\cdot E_E(1)+\left\{(1-p^{k+1})\binom{k}{1}p(1-p)^{k-1}\cdot E_E(2)\right\}\right]\right\} \qquad (2)$$

Next, *Steps 3, 4* are executed when a data packet arrives, which happens with probability *k*/(*k*+1). Again, assuming that multiple block losses are negligible, the blockID of the incoming data packet does not match expBlockID if and only if all previous data packets from the current block and the FEC packet of the previous block (i.e. blockID-1) are lost; the probability of this happening is $1/k\cdot\sum_{i=1}^{k}p^i$. Therefore, the power consumed by *Steps 3, 4* is:

$$f\left\{\frac{k}{k+1}\left[\left(1-\frac{1}{k}\sum_{i=1}^{k}p^i\right)\cdot E_E(3)+\frac{1}{k}\sum_{i=1}^{k}p^i\cdot E_E(4)\right]\right\} \qquad (3)$$

Adding Equations (2) and (3) gives an estimate of the total power consumed at the egress, P_E .

5. Results and Discussion

The energy linked to various hardware operations in Section 3 is derived from data sheets such as Intel's IXP2800 10Gbps Network Processor, SRAM [3], DRAM [4] and other chip specifications. We obtain the following values after examining them – energy/bit for an SRAM (DRAM) access is 0.168 nJ (0.092 nJ) – this includes the energy associated with memory controllers, TCAMs consume 0.967 nJ/bit, 10G transceivers (at edge routers) need 0.1 nJ to transmit/receive a bit, 2-bit operations (add/subtract etc.) incur 0.10 pJ, and XOR-ing two bits costs 0.01 pJ.

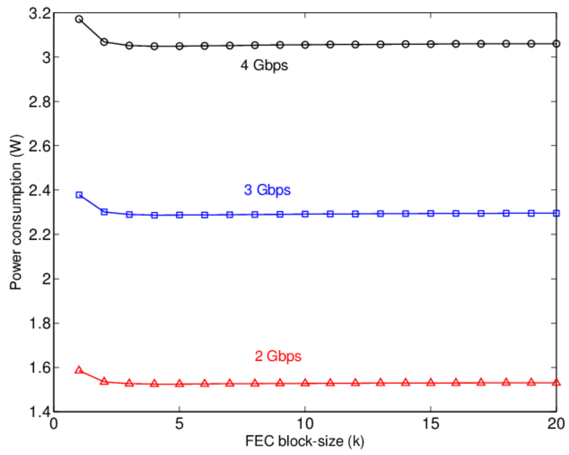

Fig. 2: Power consumption for different data rates

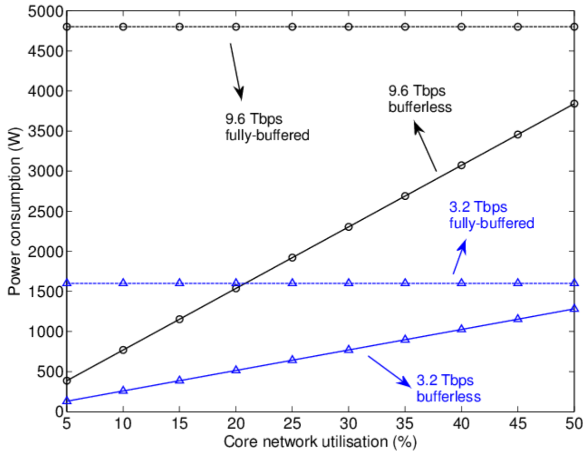

Fig. 3: Network-wide power consumption for varying capacities

Fig. 2 shows the power consumed by the FEC scheme (i.e. $P_I + P_E$ from Eqs. (1)-(3)) as a function of block-size *k* for different data rates. The packet loss rate *p* is assumed to be 10^{-2}. The size of each packet is 1000 Bytes, and the data rate (in Gbps) is increased by varying the packet rate *f*. This causes the FEC power consumption to increase as well; averaging ≈ 0.8 W/Gbps. Next, Fig. 3 depicts the network-wide power consumption with FEC (the bufferless curves) and without FEC (the fully-buffered curves) as a function of utilisation for two chosen total network capacities. The former has 30 core routers while the latter 10. Each router is assumed to be a Cisco CRS-1 with an average of 8 line-cards, and each line-card operates at 40Gbps (for a total capacity of 9.6 and 3.2Tbps, respectively).

We conservatively assume that packet buffers consume 20W (≈ 5% of a line-card's power). The important point to emerge from Fig. 3 is that when all line-cards are fully-buffered (i.e. the horizontal curve), the overall power consumption at 9.6Tbps due to packet buffers is 4.8KW (buffering energy is constant under low/moderate utilisation). On the other-hand, our FEC scheme consumes substantially less power – between 1.5-2.3 KW for 20-30% utilisation. This presents a saving of well over 50%. Similar gains follow for the second scenario (3.2Tbps) too.

6. Conclusions

We showed that our novel XOR based packet-level FEC scheme offers overall power savings by eliminating buffers from core routers and protecting data packets with FEC codes.

7. References

[1] A. Vishwanath, V. Sivaraman et al., *"Adapting Router Buffers for Energy Efficiency,"* ACM SIGCOMM CoNEXT, Dec 2011 (To appear).
[2] A. Vishwanath et al., *"Enabling a Bufferless Core Network Using Edge-to-Edge Packet-Level FEC,"* IEEE INFOCOM, Mar 2010.
[3] Cypress 4 Mbit SRAM data sheet *http://www.cypress.com/?docID=31092*
[4] Micron DDR3 SDRAM power calculator *http://download.micron.com/downloads/misc/ddr3_power_calc.xls*