A Simpler and Better Design of Error Estimating Coding

Nan Hua[1], Ashwin Lall[2], Baochun Li[3], Jun (Jim) Xu[1]

[1]School of CS, Georgia Tech [2]Dept of Math and CS, Denison Univ.
[3]Dept of ECE, Univ of Toronto

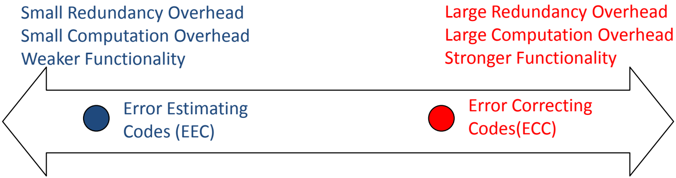
Presented in IEEE Infocom 2012,
Mar 27, 2012, Orlando, FL

Summary

- Backgrounds
  - Error Estimating Code: Definition and Motivation
  - Two-party communication and Tug-of-war sketch
- Our approach
  - Naïve Tug-of-War
  - Enhanced Tug-of-War (EToW)
  - Analytical results of EToW
  - Numerical Results of EToW

Background: Problem Definition

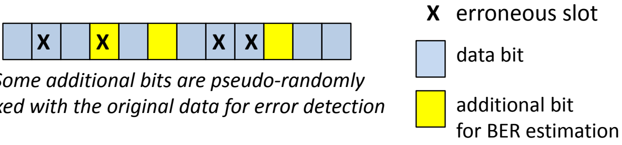
- Question:
  - Can we estimate bit error rate **directly** from the packet received?
    - ★ Not indirectly infer from packet loss ratio or signal/noise ratio
- Seminally proposed by [B.Chen, et al, SIGCOMM10]
  Motivations:
  - Not necessarily to correct every bit
  - Benefit packet re-scheduling, routing, carrier selection, wifi-rate adaptation, etc



- Our work presented here is focused on
  - Can we design a better alternative EEC code?

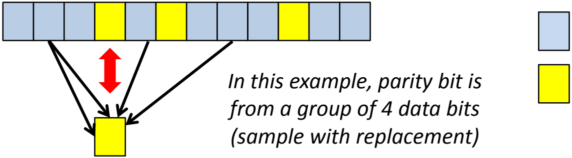
Background: How to design Error Estimating Code?

- Something should be added into the packet sent to enable BER estimation on the receiver side



Some additional bits are pseudo-randomly mixed with the original data for error detection

- How to construct those bits?
  - Authors of [B.Chen, et al, SIGCOMM10] propose to use an array of parity bits.
  - Each EEC bit is a parity bit of a group of bits sampled



In this example, parity bit is from a group of 4 data bits (sample with replacement)

Background: Original EEC design (cont'd)

- Parity bits sampled from the same group size is only suitable for detecting error rate p in a certain range.
  - if p is too high or too low, what would happen?
- A multi-level design to estimate p in a wide range proposed in [B.Chen, et al, SIGCOMM10]
  - The sampling group size of each level is geometrically distributed: 2,4,8,...
  - Asymptotically, $O(\log n)$ bits are needed.
  - Typically, the authors suggest a 9-level 32-bits-per-level design to estimate p in $[0.001, 0.15]$

Our approach: From the angle of two-party communication

- Two-party communication
  - Bob knows (local) string x; Alice knows (local) string y
  - How to use (minimum amount of) communication to compute $f(x, y)$ (maybe approximately)?
  - The minimum amount is referred to as *communication complexity*
- Casted to the problem here:
  - One-round two-party communication problem
  - Target function f is the hamming distance $f(x, y) = \|x - y\|_0$.
- Any benefit from this angle?
  - Leverage the rich results in two-party communication

Our approach: From the angle of two-party communication(cont'd)

- $\Omega(\frac{1}{\epsilon^2}\log n)$ bits are needed
  - in order to approximately compute $\|x - y\|_0$ by less than ϵ error w.h.p.
  - randomization and approximation are both proved to be required
  - Proof can be found in our paper
- Therefore, the original design is already the best asymptotically!!
- Can we improve the design from a practical perspective?
  - Leverage an established result: Tug-of-war Sketch
    - (originally designed for estimating L_2 norm in data streaming application)
  - Here comes our Enhanced Tug-of-War sketch (EToW) for EEC problem.

Key idea of the Tug-of-war Sketch

- Established for the two-party computation of L_2 norm.
- Key Ideas:
  - Random project x and y by a pre-defined pseudorandom vector $\vec{s} \in \{-1, 1\}^n$
    - ★ calculate inner product $\vec{x} \cdot \vec{s}$ and $\vec{y} \cdot \vec{s}$.
    - ★ Note: those inner products are only $\log n$ bits.
  - Use $\|\vec{x} \cdot \vec{s} - \vec{y} \cdot \vec{s}\|_2$ to approximate $\|\vec{x} - \vec{y}\|_2$
  - Note:
    - ★ x and y are both binary,
      hence measuring $\|\vec{x} - \vec{y}\|_2 \Leftrightarrow$ measuring $\|\vec{x} - \vec{y}\|_0$
    - ★ $\vec{x} \cdot \vec{s}$ for binary data is actually equivalent to an xor with a random binary string and a pop-count operation. In practice, we use $\#1 - \frac{1}{2}l$ of the result, which is equal to $\vec{x} \cdot \vec{s}/2$.
      (for convenience, we define the binary $\vec{x}$ and $\vec{y}$ in $\{-1, 1\}^n$ in the inner product)

Tug-of-war Sketch for EEC

SKETCH–CREATION($\vec{b}$)

Input $\vec{b}$: original data bits vector.
Output z: the sketch encoding $\vec{b}$.
pre-compute random vectors $\vec{s}_{j,1\le j\le c} : [n] \to \{-1, 1\}$
for $j = 1$ to c **do**
$z_j := (\vec{b} \cdot \vec{s}_j)/2$
return $z = \langle z_1, \ldots, z_c \rangle$

DISTANCE–ESTIMATION($\vec{b'}, z$)

Input $\vec{b'}$: received data bits vector, z: received sketch.
Output $\hat{p}$: the estimated error rate.
pre-compute random vectors $\vec{s}_{j,1\le j\le c} : [n] \to \{-1, 1\}$
for $j = 1$ to c **do**
$X_j := (z_j - \vec{b'} \cdot \vec{s}_j/2)^2$
return $\hat{p} = \frac{1}{n}$average$(X_1, \ldots, X_c)$

Note that $E[X_j] = E[(\frac{1}{2}(\vec{b} - \vec{b'}) \cdot \vec{s}_j)^2] = p$

However

- No advantage in size compared to the original design of EEC.
  - Suppose the packet length is 1500bytes$=12,000$bits. Each counter need to be as long as 14 bits.
  - Suppose we need 16 counters
  - 16×14 already similar to the cost of the original design of EEC.
- Moreover, errors inside each sketch might totally corrupt the approximation!
  - An immediate remedy: do "error correction" on those bits
  - Even more overhead...
- Conclusion: Naïve Tug-of-war Sketch is not a good fit for EEC.

Key ideas of the Enhanced Tug-of-War Sketch (EToW)

- Not necessary to "fully correct" the sketch with errors inside
  - Just need to detect the corrupted counter value(s)
  - Use one or two additional parity bit(s) to check
- From streaming vs. sampling perspective, streaming (every bit participates in the calculation) not necessarily better than sampling
  - if we first sample l bits to build the sketch from the sampled bits
  - length of each counter reduced from $\log_2(n)$ to $\log_2(l)$.
- the higher bits of the counter value ($\frac{1}{2}(\vec{b} \cdot \vec{s}_j)$) are not as informative as the lower bits
  - It's a sum of random -1 and 1s and concentrated around 0. Highest bits are w.h.p 0.
  - simply use the lower k bits, k could be as small as 5.
    - ★ The substraction in $X_j = (z_j - \vec{b'} \cdot \vec{s}_j/2)^2$ defined on F_{2^k} field (the same as the standard substraction of integer in computers)
    - ★ Small overflow only slightly influence the result

Enhanced Tug-of-War sketch (Only additional steps listed)

SKETCH–CREATION($\vec{b}$)

for $j = 1$ to c **do**

Random projection: $\tilde{z}_j := (\vec{b}_j \cdot \vec{s}_j)/2$

k-bits-long truncated projection: $z_j := \mathrm{trunc_k}(\tilde{z}_j)$

r-bits-long parities $q_j := \mathrm{parity_r}(z_j)$

return $c(k + r)$-bits long sketch $z = \langle z_1, \ldots, z_c\rangle\langle q_1, \ldots, q_c\rangle$

DISTANCE–ESTIMATION($\vec{b}', \check{z}$)

for $j = 1$ to c **do**

Random projection: $\tilde{z}_j{}' := (\vec{b}'_j \cdot \vec{s}_j)/2$

Estimation $Y_j := \mathrm{trunc_k}(\tilde{z}_j{}' - \check{z}_j)$,$X_j = Y_j^2$

Check parities $V_j := 1_{\{q_j = \mathrm{parity_r}(\mathrm{trunc_k}(\tilde{z}_j{}'))\}}$

return $\hat{p} = \frac{\sum_{j=1}^{c} V_j X_j}{l \sum_{j=1}^{c} V_j}$ as the estimation of error rate p.

Analysis of Enhanced Tug-of-War Sketch

- Although the three techniques (sampling, truncation, parity checking) we used in EToW looks heuristic, their impacts are still analyzable.
- Welcome to our paper to find the details of analysis. Here we will only highlight the following conclusions drawn from the analysis.

Impact of Sampling Parameter l

- Use sampling to build tug-of-war sketch would lead to rising relative errors in the smaller p region.

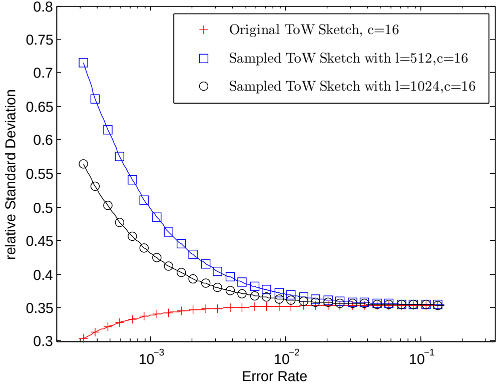


Figure: Sampled and Original tug-of-war sketches with $c = 16$.

Impact of Truncation Parameter k

- Small k (more truncation) will lead to rising relative errors in the larger p region.

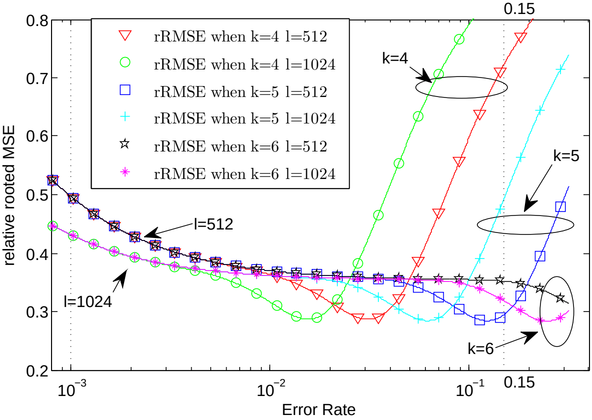
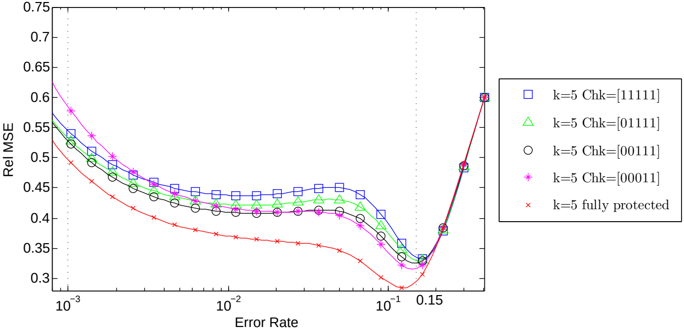


Figure: EToW's relMSE (full protection assumed) with different parameters: $c = 16, l = \{512, 1024\}, k = \{4, 5, 6\}$.

Impact of Errors on the sketch

- Adjust the generation matrix of the parity checking bit to see the impact.



- $[11111]$ means parity checking all bits in the 5-bit counter
  $[00111]$ means only checking the higher 3 bits in the 5-bit counter

Numerical Results

- Metrics:
  - relative mean squared error
  - ratio of large errors
  - tail probability distribution
- Candidates in the comparison:
  - Original EEC
    - $\hat{p_1}$ and $\hat{p_2}$ are the two estimators of the original EEC design with 9-level 32-bit-per-level.
  - ETOW
    - 48 6-bit counters (c=48): same size
    - 16 6-bit counters (c=16): 1/3 size, similar performance

Numerical Result: mean squared error

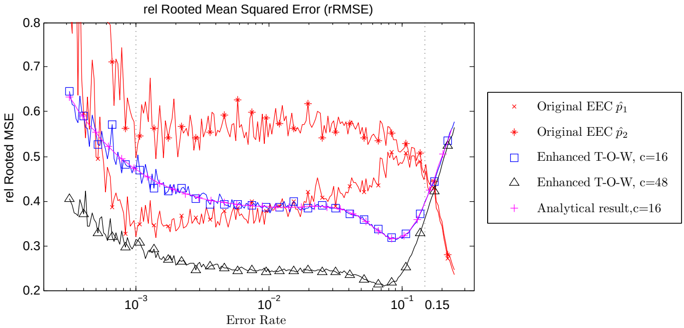


Figure: Relative MSE of different schemes

Numerical Result:Ratio of large errors

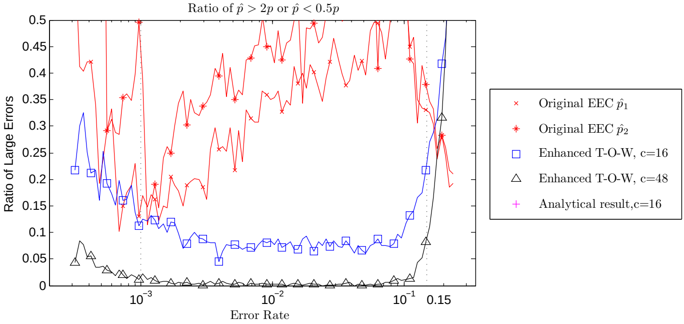


Figure: Ratio of large errors ($\hat{p} > 2p$ or $\hat{p} < \frac{1}{2}p$ of different schemes)

Numerical Result: Tail distribution

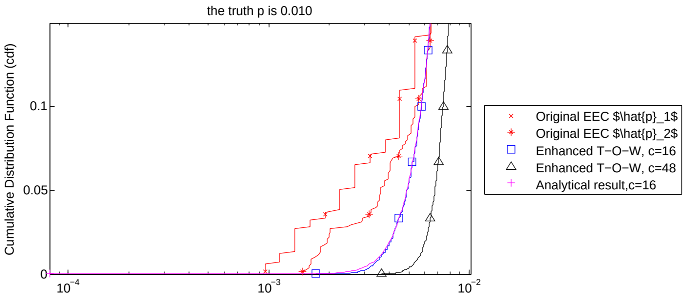


Figure: An example of tail distribution of different schemes

Conclusion

- Re-visited the design of error-estimating coding problem from a different angle
- Proved the original EEC design is already asymptotically optimal
- Proposed enhanced tug-of-war sketch which is better in practical for wide-range BER estimation

Thanks for your questions!

- My contact: nanhua@gatech.edu