Rateless Erasure Codes

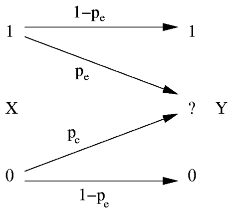
Karthik GVB, Fathima Zarin Faizal

Department of Electrical Engineering
IIT Bombay

Outline

The BEC

Consider the binary erasure channel:

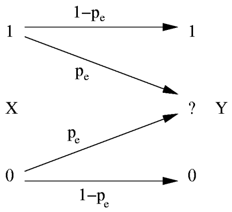


BEC [1]

[1]Image source: Wikipedia

The BEC

Consider the binary erasure channel:



BEC [1]

For example, the Internet.

[1] Image source: Wikipedia

Reliable transmission over BEC

TCP/IP

Reliable transmission over BEC

TCP/IP

- Do we really need a feedback channel?

Reliable transmission over BEC

TCP/IP

- Do we really need a feedback channel? No.

Reliable transmission over BEC

TCP/IP

- Do we really need a feedback channel? No.
- Sending Mars pics back to Earth using this?

Reliable transmission over BEC

TCP/IP

- Do we really need a feedback channel? No.
- Sending Mars pics back to Earth using this?
  If you want to wait that long.

Reliable transmission over BEC

TCP/IP

- Do we really need a feedback channel? No.
- Sending Mars pics back to Earth using this?
  If you want to wait that long.
- Broadcast channels?

Reliable transmission over BEC

TCP/IP

- Do we really need a feedback channel? No.
- Sending Mars pics back to Earth using this?
  If you want to wait that long.
- Broadcast channels? Dead.

Reliable transmission over BEC

TCP/IP

- Do we really need a feedback channel? No.
- Sending Mars pics back to Earth using this?
  If you want to wait that long.
- Broadcast channels? Dead.

This calls for some coding theory rescue.

What to expect from an erasure correcting code

What to expect from an erasure correcting code

- Recover info despite lost bits

What to expect from an erasure correcting code

- Recover info despite lost bits
- Correct as many erasures as possible

What to expect from an erasure correcting code

- Recover info despite lost bits
- Correct as many erasures as possible
- (Almost (?)) no feedback

What to expect from an erasure correcting code

- Recover info despite lost bits
- Correct as many erasures as possible
- (Almost (?)) no feedback
- Fast algorithms

Reed Solomon Codes

Consider (N, K) RS codes over F_{2^l}.

Reed Solomon Codes

Consider (N, K) RS codes over F_{2^l}.

Advantages

- Needs just K/N transmitted symbols

Reed Solomon Codes

Consider (N, K) RS codes over F_{2^l}.

Disadvantages

- Require small N, K, q

Reed Solomon Codes

Consider (N, K) RS codes over F_{2^l}.

Disadvantages

- Require small N, K, q
- $K(N-K)\log_2 N$ packet operations

Reed Solomon Codes

Consider (N, K) RS codes over F_{2^l}.

Disadvantages

- Require small N, K, q
- $K(N-K)\log_2 N$ packet operations
- Need to estimate p_e

Reed Solomon Codes

Consider (N, K) RS codes over F_{2^l}.

Disadvantages

- Require small N, K, q
- $K(N-K)\log_2 N$ packet operations
- Need to estimate p_e

What if estimated p_e is lesser?

Reed Solomon Codes

Consider (N, K) RS codes over F_{2^l}.

Disadvantages

- Require small N, K, q
- $K(N-K)\log_2 N$ packet operations
- Need to estimate p_e

What if estimated p_e is lesser?

Enter rateless erasure codes.

A digital fountain

A digital fountain

The Encoder [2]

A digital fountain

Water drops $\equiv$ encoded packets

A digital fountain

Water drops $\equiv$ encoded packets

- Source file: $K\ell$ bits

A digital fountain

Water drops $\equiv$ encoded packets

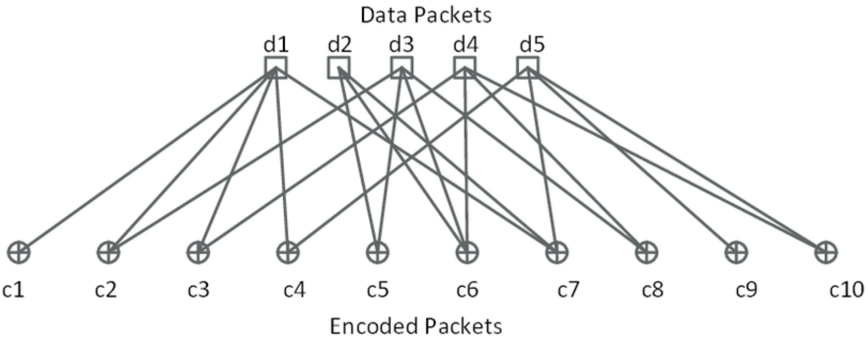
- Source file: $K\ell$ bits
- Water drop: ℓ encoded bits

A digital fountain

Water drops $\equiv$ encoded packets

- Source file: $K\ell$ bits
- Water drop: ℓ encoded bits
- Collect $\approx K$ drops

Fountain code encoder

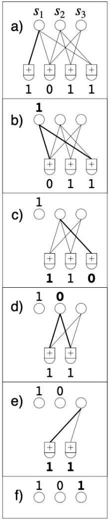


The encoding process [2]

[2] Image source: https://www.researchgate.net/figure/A-typical-encoding-process-of-LT-codes_fig2_307915267

Fountain code decoder

Complexity of decoding algorithm determined by #edges in this graph (Image source: [1])



$(K, \mathcal{D})$ fountain code

$(K, \mathcal{D})$ fountain code

- Linear map from $F_{2^K} \rightarrow F_{2^{\mathbb{N}}}$

$(K, \mathcal{D})$ fountain code

- Linear map from $F_{2^K} \to F_{2^{\mathbb{N}}}$
- $x \in F_{2^K}, y \in F_{2^{\mathbb{N}}}$

$(K, \mathcal{D})$ fountain code

- Linear map from $F_{2^K} \rightarrow F_{2^{\mathbb{N}}}$
- $x \in F_{2^K}, y \in F_{2^{\mathbb{N}}}$
- Coordinates are independent rvs generated using the dist. $\mathcal{D}$

$(K, \mathcal{D})$ fountain code

- Linear map from $F_{2^K} \to F_{2^{\mathbb{N}}}$
- $x \in F_{2^K}, y \in F_{2^{\mathbb{N}}}$
- Coordinates are independent rvs generated using the dist. $\mathcal{D}$

To generate an output symbol y_i:

$(K, \mathcal{D})$ fountain code

- Linear map from $F_{2^K} \to F_{2^{\mathbb{N}}}$
- $x \in F_{2^K}, y \in F_{2^{\mathbb{N}}}$
- Coordinates are independent rvs generated using the dist. $\mathcal{D}$

To generate an output symbol y_i:

1. Sample $\mathcal{D}$ to obtain a weight w from 1 to K

$(K, \mathcal{D})$ fountain code

- Linear map from $F_{2^K} \to F_{2^{\mathbb{N}}}$
- $x \in F_{2^K}, y \in F_{2^{\mathbb{N}}}$
- Coordinates are independent rvs generated using the dist. $\mathcal{D}$

To generate an output symbol y_i:

1. Sample $\mathcal{D}$ to obtain a weight w from 1 to K
2. $v \in F_{2^K}$ of weight w is chosen uniformly at random

$(K, \mathcal{D})$ fountain code

- Linear map from $F_{2^K} \to F_{2^{\mathbb{N}}}$
- $x \in F_{2^K}, y \in F_{2^{\mathbb{N}}}$
- Coordinates are independent rvs generated using the dist. $\mathcal{D}$

To generate an output symbol y_i:

1. Sample $\mathcal{D}$ to obtain a weight w from 1 to K
2. $v \in F_{2^K}$ of weight w is chosen uniformly at random
3. $y_i = \sum_j v_j x_j$

Setting up the problem

- A **reliable decoding algorithm** of length N for a Fountain code is an algorithm which can recover the K input symbols from any set of output symbols and errs with a probability that is at most $1/K$

Setting up the problem

- A **reliable decoding algorithm** of length N for a Fountain code is an algorithm which can recover the K input symbols from any set of output symbols and errs with a probability that is at most $1/K$
- **Cost** $=$ Expected #arithmetic operations

Setting up the problem

- A **reliable decoding algorithm** of length N for a Fountain code is an algorithm which can recover the K input symbols from any set of output symbols and errs with a probability that is at most $1/K$
- **Cost** $=$ Expected #arithmetic operations

Design objective: $\mathcal{D}$ should enable simple linear time decoding of $\{x_1, \ldots, x_K\}$ as soon as any $K(1+\epsilon)$ of y-s are received.

Lower bound on cost

Lower bound on cost

Proposition

If an LT-Code with K input symbols has a reliable decoding algorithm, then there is a constant c such that the associated graph has at least $cK\ln(K)$ edges.

Lower bound on cost

Proposition

If an LT-Code with K input symbols has a reliable decoding algorithm, then there is a constant c such that the associated graph has at least $cK\ln(K)$ edges.

Proof: By reliable decoding, we mean that error probability is at most $1/K^u$(u is some constant)

Lower bound on cost

Proposition

If an LT-Code with K input symbols has a reliable decoding algorithm, then there is a constant c such that the associated graph has at least $cK\ln(K)$ edges.

Proof: By reliable decoding, we mean that error probability is at most $1/K^u$(u is some constant)

Let the LT-Code has the degree distribution $\rho(d)$ and let $\mathcal{G}$ denotes decoding graph(a bipartite graph with K input nodes and N output nodes).

Lower bound on cost

(*Continuation*)

Consider an output node and degree d is chosen, then for any input node ν in $\mathcal{G}$, the probability that the input node ν is not a neighbour of the output node is $1 - d/K$

Lower bound on cost

(*Continuation*)

Consider an output node and degree d is chosen, then for any input node ν in $\mathcal{G}$, the probability that the input node ν is not a neighbour of the output node is $1 - d/K$

In general, the probability that the input node ν is not a neighbour of an output node is

$$\sum_d \rho(d) \cdot (1 - d/K) = 1 - a/K$$

Lower bound on cost

(*Continuation*)

Consider an output node and degree d is chosen, then for any input node ν in $\mathcal{G}$, the probability that the input node ν is not a neighbour of the output node is $1 - d/K$

In general, the probability that the input node ν is not a neighbour of an output node is

$$\sum_d \rho(d) \cdot (1 - d/K) = 1 - a/K$$

where a is the average degree of an output node, and so the probability that ν is not a neighbour any of the output nodes is $(1 - a/K)^N$.

Lower bound on cost

(*Continuation*)

Consider an output node and degree d is chosen, then for any input node ν in $\mathcal{G}$, the probability that the input node ν is not a neighbour of the output node is $1 - d/K$

In general, the probability that the input node ν is not a neighbour of an output node is

$$\sum_d \rho(d) \cdot (1 - d/K) = 1 - a/K$$

where a is the average degree of an output node, and so the probability that ν is not a neighbour any of the output nodes is $(1 - a/K)^N$.
By Taylor expansion of $ln(1 - x)$ gives,

$$ln(1 - a/K) \geq (a/K)/(1 - a/K) \implies (1 - a/K)^N \geq e^{-\alpha/(1-\alpha/n)}$$

where $\alpha = aN/K$

Lower bound on cost

(*Continuation*)

Now, $(1 - a/K)^N \leq 1/K^u$ as probability of error is lower bounded by the probability that there is an uncovered node,

Lower bound on cost

(*Continuation*)

Now, $(1 - a/K)^N \leq 1/K^u$ as probability of error is lower bounded by the probability that there is an uncovered node,

$$
\begin{aligned}
e^{-\alpha/(1-\alpha/n)} &\leq 1/K^u \\
\implies \alpha &\geq ln(K)\frac{u}{1 + uln(K)/N} \\
&\geq cln(K)
\end{aligned}
$$

where $c = u/(log(2)(1 + uln(3)/3))$

Lower bound on cost

(*Continuation*)

Now, $(1 - a/K)^N \leq 1/K^u$ as probability of error is lower bounded by the probability that there is an uncovered node,

$$
\begin{aligned}
e^{-\alpha/(1-\alpha/n)} &\leq 1/K^u \\
\implies \alpha &\geq ln(K)\frac{u}{1 + uln(K)/N} \\
&\geq cln(K)
\end{aligned}
$$

where $c = u/(log(2)(1 + uln(3)/3))$

$$aN \geq cKlog(K)$$

aN is the average number of edges in the graph $\mathcal{G}$. $\square$

Degree distributions

Degree distributions

Ideal Soliton distribution

$\rho(d)$ for $d = \{1, 2, \cdots, K\}$

- $\rho(1) = 1/K$
- For all $i = 2, \cdots, K$, $\rho(i) = 1/i(i-1)$.

Degree distributions

Ideal Soliton distribution

$\rho(d)$ for $d = \{1, 2, \cdots, K\}$

- $\rho(1) = 1/K$
- For all $i = 2, \cdots, K,\ \rho(i) = 1/i(i-1)$.

- Average degree $\approx ln(K)$ which implies that the sum of degrees of K encoding symbols is on average $Kln(K)$

Degree distributions

Ideal Soliton distribution

$\rho(d)$ for $d = \{1, 2, \cdots, K\}$

- $\rho(1) = 1/K$
- For all $i = 2, \cdots, K, \; \rho(i) = 1/i(i-1)$.

- Average degree $\approx ln(K)$ which implies that the sum of degrees of K encoding symbols is on average $Kln(K)$
- Is good only in an expected sense

Degree distributions

Robust Soliton distribution

$\mu(\cdot)$, Let $R = cln(K/\delta)\sqrt{K}$, Define $\tau(\cdot)$ as follows,

$$\tau(i) = \begin{cases} R/ik & \text{for } i = 1, \cdots, K/R - 1 \\ Rln(R/\delta)/K & \text{for} i = K/R \\ 0 & \text{for} i = K/R + 1, \cdots, K \end{cases}$$

Add the Ideal Soliton distribution $\rho(\cdot)$ to $\tau(\cdot)$ and normalize to obtain $\mu(\cdot)$:

- $\beta = \sum_{i=1}^{K} \rho(i) + \tau(i)$

Degree distributions

Robust Soliton distribution

$\mu(\cdot)$, Let $R = cln(K/\delta)\sqrt{K}$, Define $\tau(\cdot)$ as follows,

$$\tau(i) = \begin{cases} R/ik & \text{for } i = 1, \cdots, K/R - 1 \\ Rln(R/\delta)/K & \text{for} i = K/R \\ 0 & \text{for} i = K/R + 1, \cdots, K \end{cases}$$

Add the Ideal Soliton distribution $\rho(\cdot)$ to $\tau(\cdot)$ and normalize to obtain $\mu(\cdot)$:

- $\beta = \sum_{i=1}^{K} \rho(i) + \tau(i)$
- For all $i = 1, \cdots, K,\ \mu(i) = (\rho(i) + \tau(i))/\beta$

Degree distributions

Robust Soliton distribution

$\mu(\cdot)$, Let $R = cln(K/\delta)\sqrt{K}$, Define $\tau(\cdot)$ as follows,

$$\tau(i) = \begin{cases} R/ik & \text{for } i = 1, \cdots, K/R - 1 \\ Rln(R/\delta)/K & \text{for} i = K/R \\ 0 & \text{for} i = K/R + 1, \cdots, K \end{cases}$$

Add the Ideal Soliton distribution $\rho(\cdot)$ to $\tau(\cdot)$ and normalize to obtain $\mu(\cdot)$:

- $\beta = \sum_{i=1}^{K} \rho(i) + \tau(i)$
- For all $i = 1, \cdots, K$, $\mu(i) = (\rho(i) + \tau(i))/\beta$

- δ is the allowable failure probability
- Average degree $\approx ln(K/\delta)$

The problem with LT codes

The problem with LT codes

- LT codes have complexity at least $K \ln K$

The problem with LT codes

- LT codes have complexity at least $K \ln K$
- Can we recover all data in linear time from $\mathcal{O}(K)$ code symbols?

The problem with LT codes

- LT codes have complexity at least $K \ln K$
- Can we recover all data in linear time from $\mathcal{O}(K)$ code symbols?
  No, as this is a probabilistic model

The problem with LT codes

- LT codes have complexity at least $K \ln K$
- Can we recover all data in linear time from $\mathcal{O}(K)$ code symbols?
  No, as this is a probabilistic model
  But can recover a large fraction! How much?

The problem with LT codes

Essentially T balls (edges) thrown into K bins (i/p symbols) independently and uniformly

The problem with LT codes

Essentially T balls (edges) thrown into K bins (i/p symbols) independently and uniformly
Prob. of an empty bin $= \left(1 - \frac{1}{K}\right)^T \approx e^{-T/K}$

The problem with LT codes

Essentially T balls (edges) thrown into K bins (i/p symbols) independently and uniformly
Prob. of an empty bin $= \left(1 - \frac{1}{K}\right)^T \approx e^{-T/K}$
Expected fraction not covered $= Ke^{-T/K}/K = e^{-T/K}$

The problem with LT codes

Essentially T balls (edges) thrown into K bins (i/p symbols) independently and uniformly

Prob. of an empty bin $= \left(1 - \frac{1}{K}\right)^T \approx e^{-T/K}$

Expected fraction not covered $= Ke^{-T/K}/K = e^{-T/K}$

Hence can recover a large fraction of the data!

Raptor codes: An easy fix

Raptor codes: An easy fix

- Use weaker LT code with small average degree $\overline{d}$

Raptor codes: An easy fix

- Use weaker LT code with small average degree $\overline{d}$
- Expected unrecoverable fraction $f \approx e^{-\overline{d}}$

Raptor codes: An easy fix

- Use weaker LT code with small average degree $\overline{d}$
- Expected unrecoverable fraction $f \approx e^{-\overline{d}}$
- Use $(\frac{K}{1-f}, K)$ erasure correcting block code as outer code

Raptor codes: An easy fix

- Use weaker LT code with small average degree $\overline{d}$
- Expected unrecoverable fraction $f \approx e^{-\overline{d}}$
- Use $(\frac{K}{1-f}, K)$ erasure correcting block code as outer code

The outer code just needs to be able to correct erasures with an erasure rate of f

Raptor codes: An example

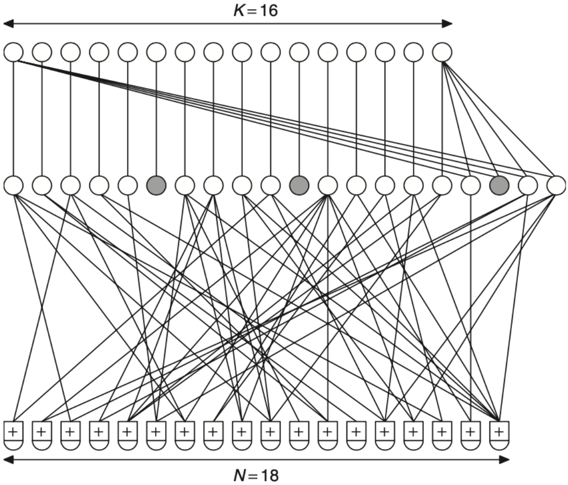
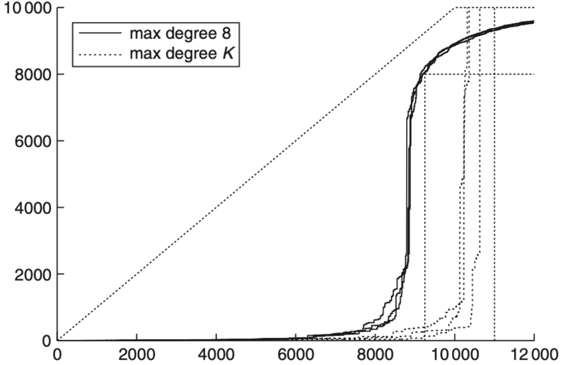


Image source: [1]

Performance comparison



#packets decoded vs #received packets [4]

[4] [1]

Summary

Summary

- Performance of fountain codes on BEC $\gg$ block codes, TCP/IP

Summary

- Performance of fountain codes on BEC $\gg$ block codes, TCP/IP
- Useful in storage and broadcast applications

References

[1] D.J.C. MacKay.
Fountain Codes.
IEE Proceedings online no. 20050237, 2005.

[2] Michael Luby.
LT Codes.
The 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002. Proceedings., 54:271–282, 2002.

[3] Emina Soljanin.
Raptor codes: From a math idea to LTE eMBMS.
2015.

[4] Amin Shokrollahi.
Raptor Codes.
IEEE Transactions on Inforamtion Theory, 52(6), 2006.

Thank you