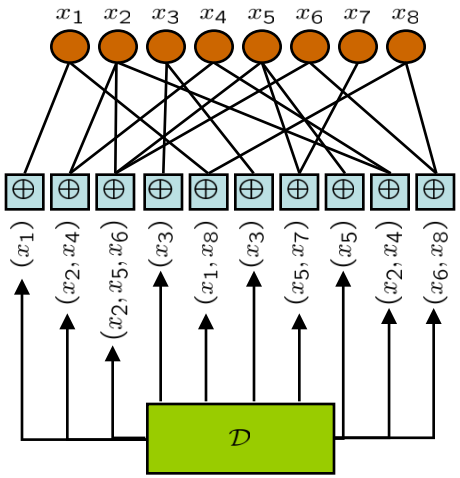
algolma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

Threshold Phenomena and Fountain Codes



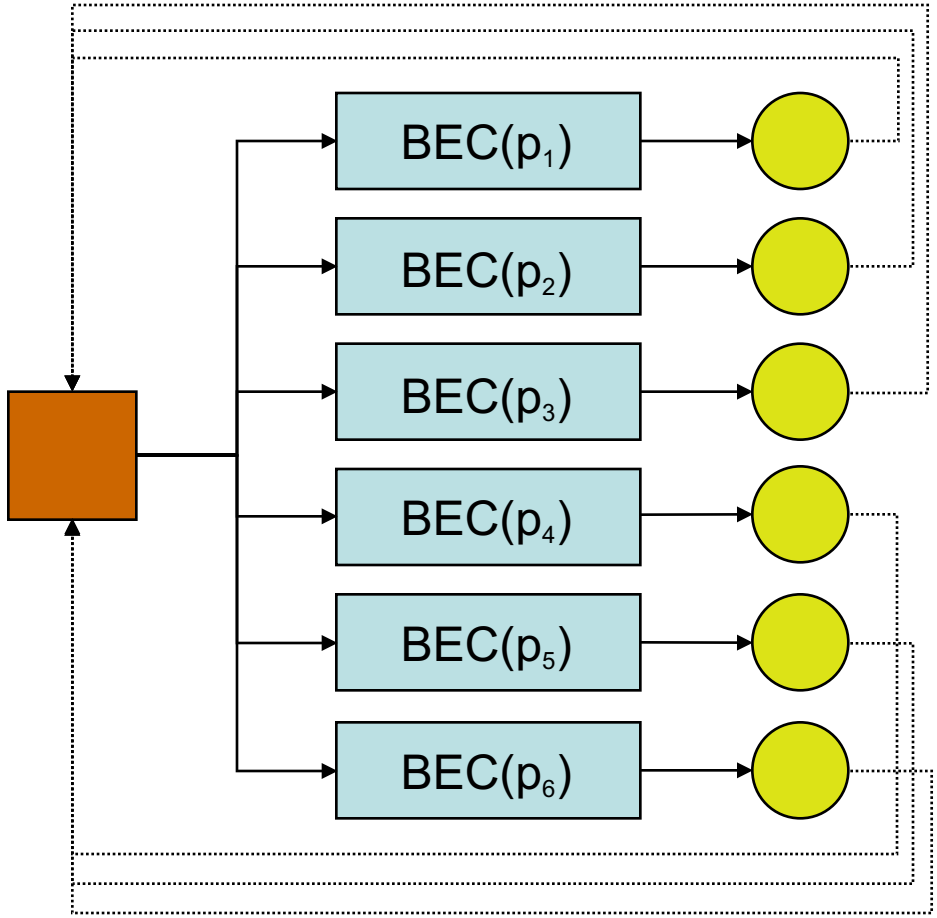
Amin Shokrollahi

EPFL

Parts are joint work with M. Luby, R. Karp, O. Etesami

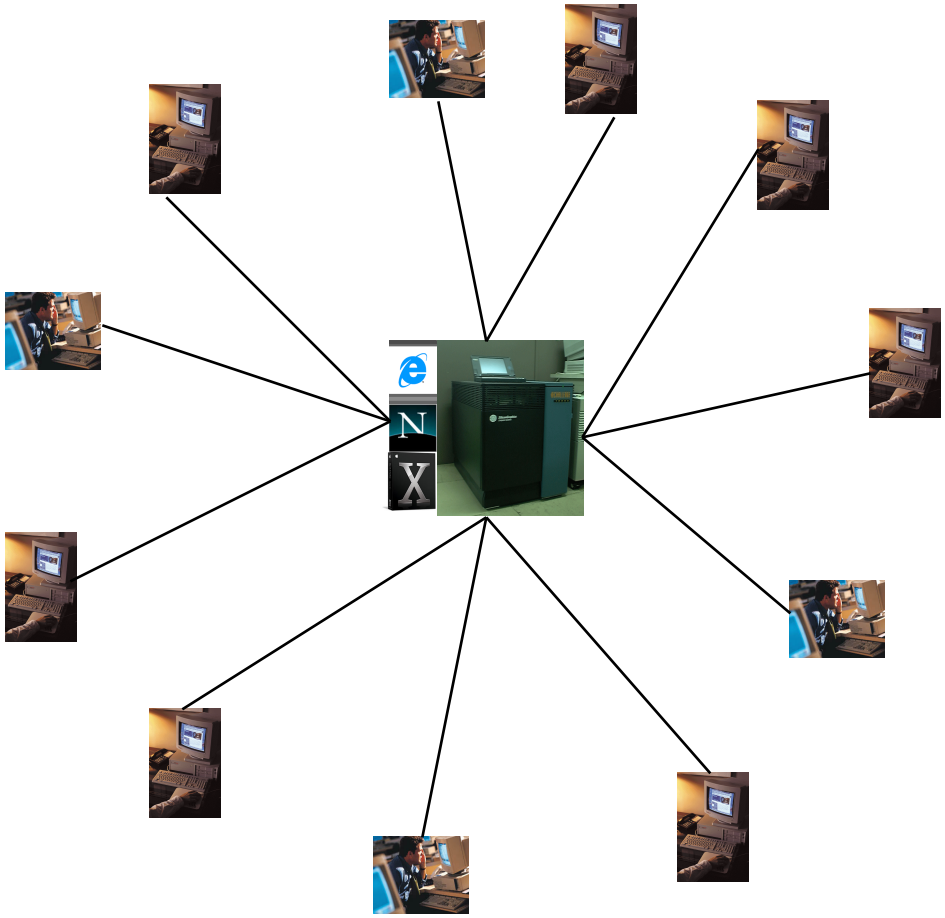
algolma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

Communication on Multiple Unknown Channels



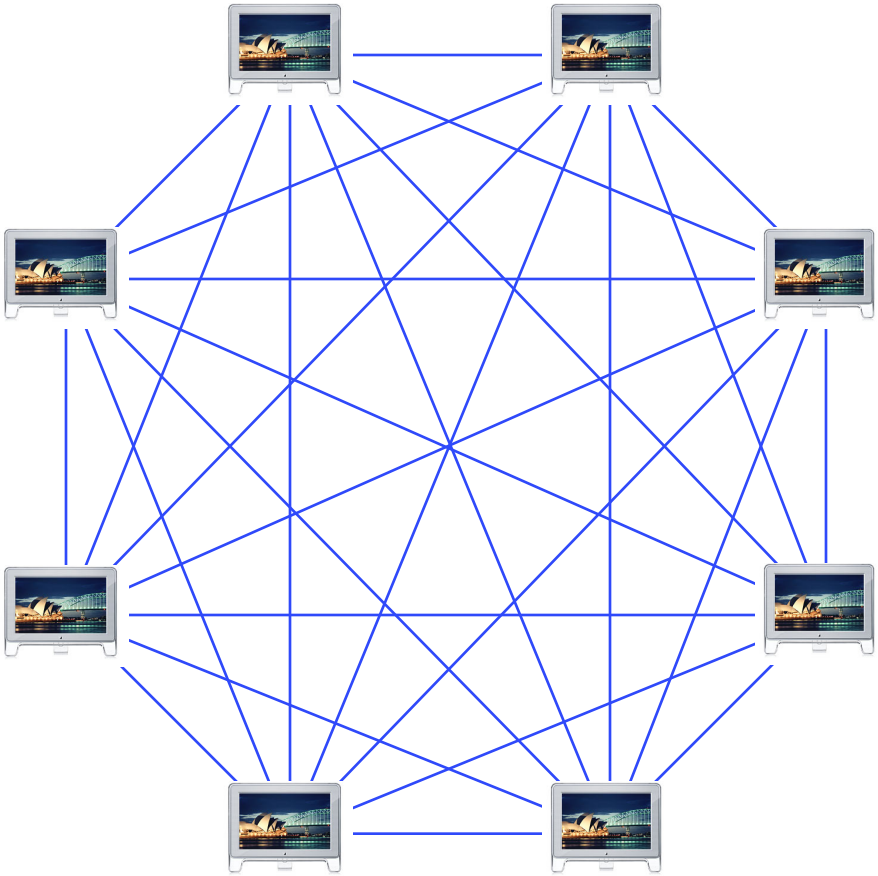
algolma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

Example: Popular Download

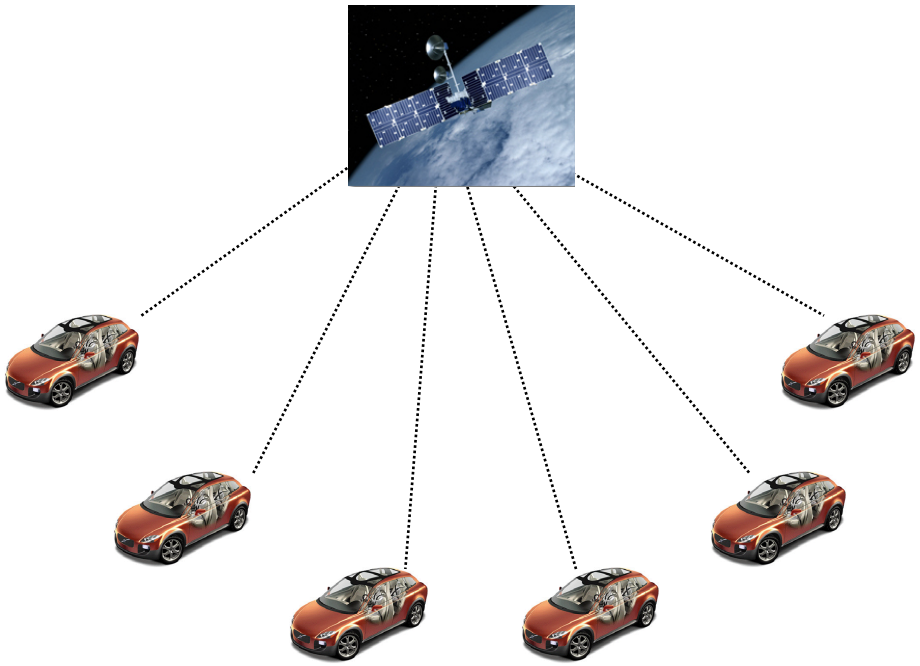


algolma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

Example: Peer-to-Peer



Example: Satellite



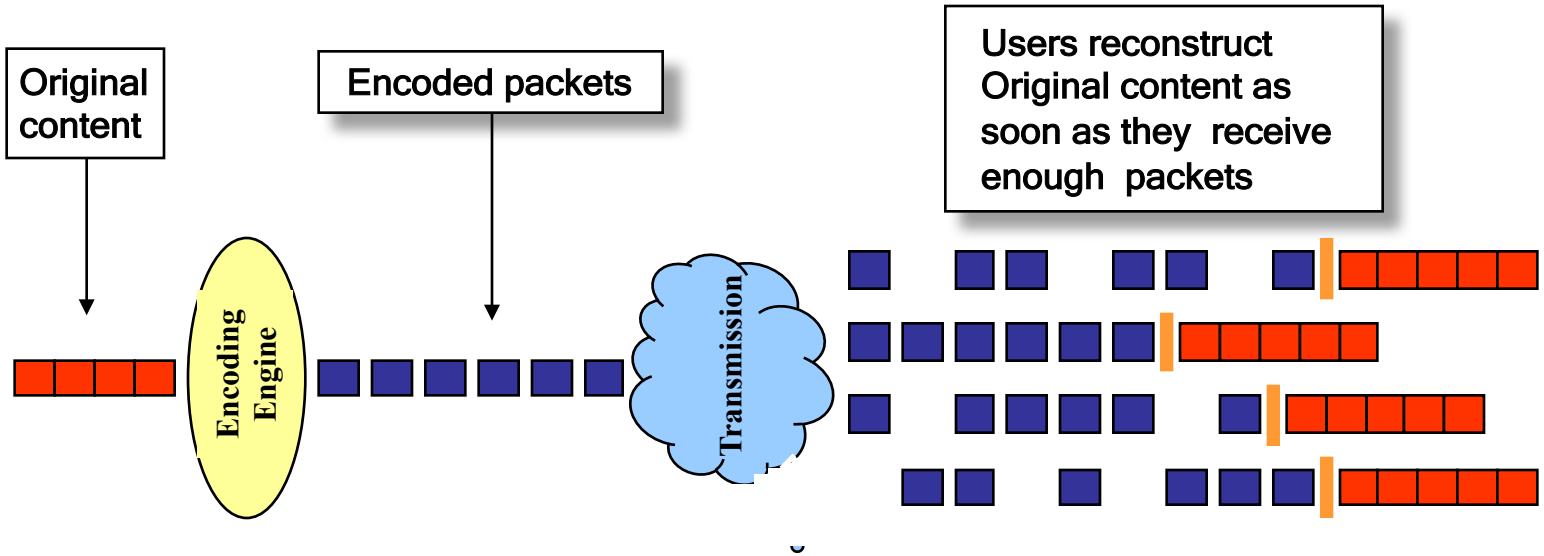
The erasure probabilities are unknown.

Want to come arbitrarily close to capacity on each of the erasure channels, with minimum amount of feedback.

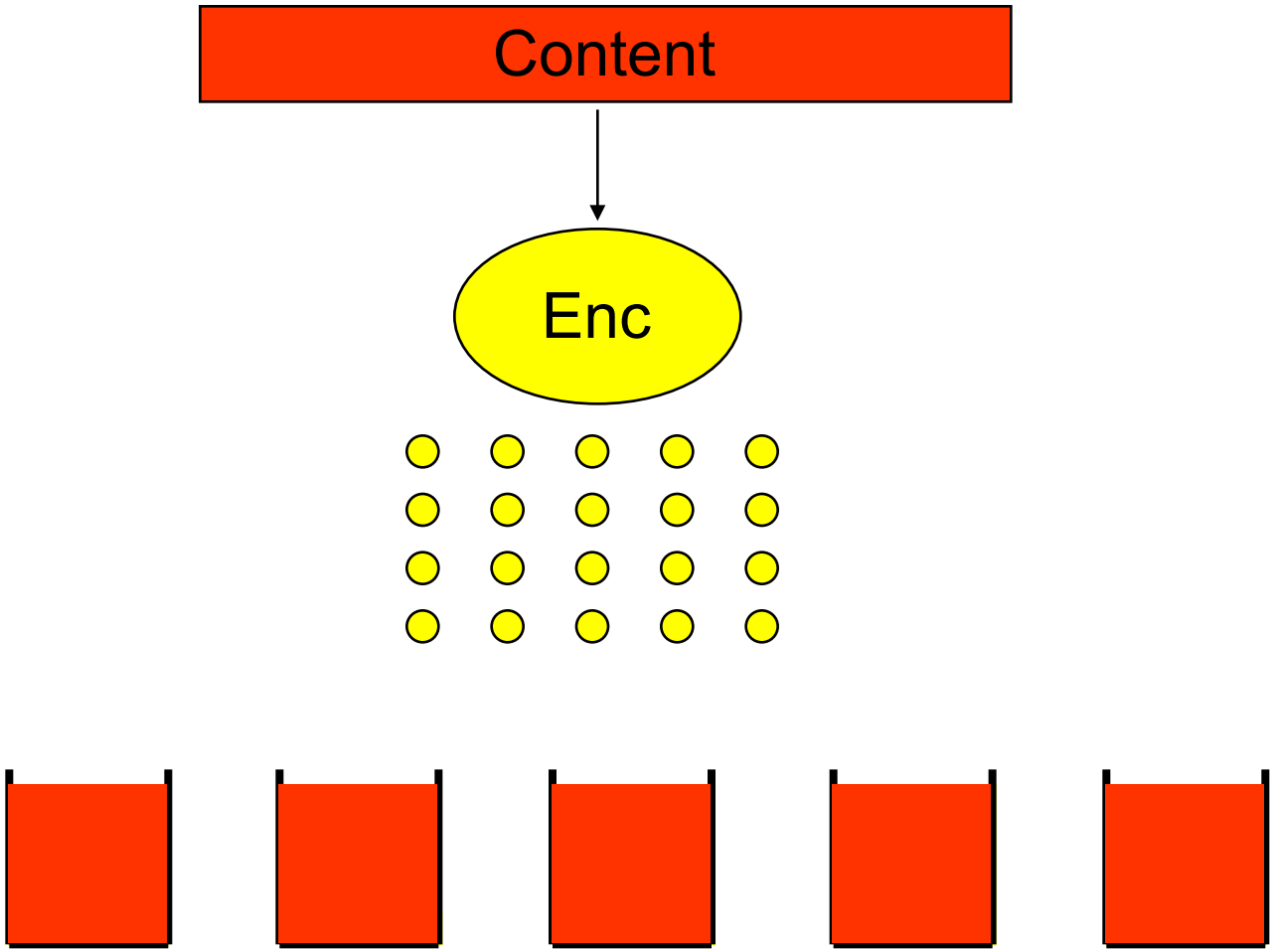
Traditional codes don't work in this setting since their rate is fixed.

Need codes that can adapt automatically to the erasure rate of the channel.

What we Really Want

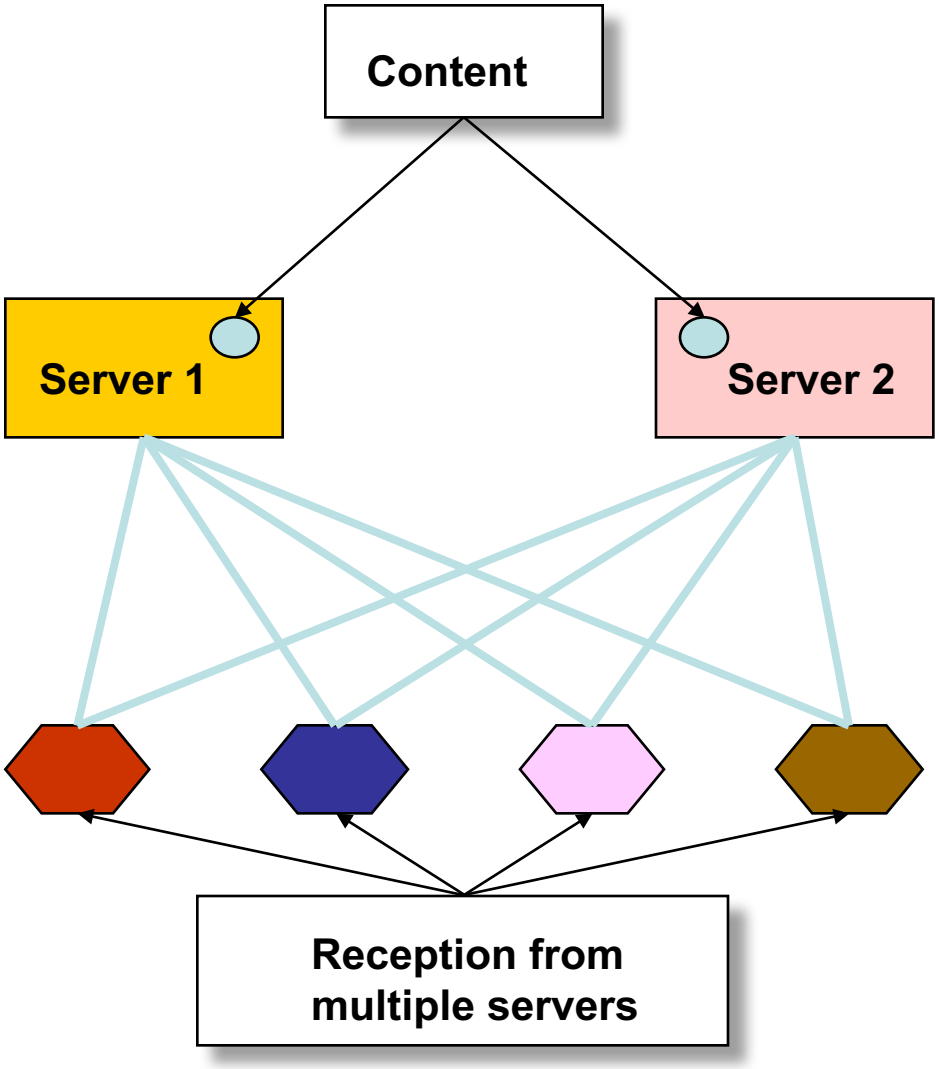


Reconstruction time should depend only on size of content

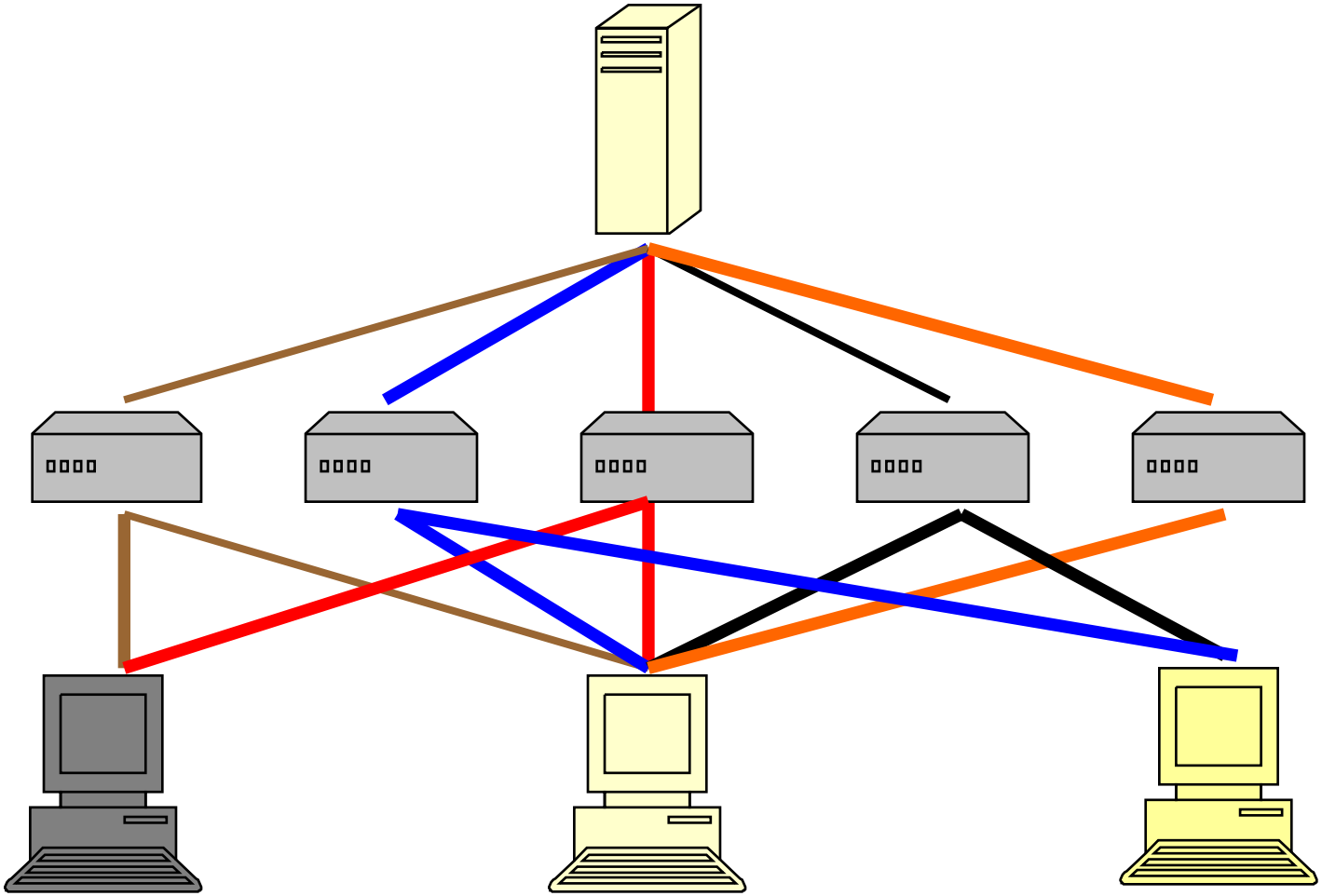


Digital buckets

Applications: Multi-site downloads

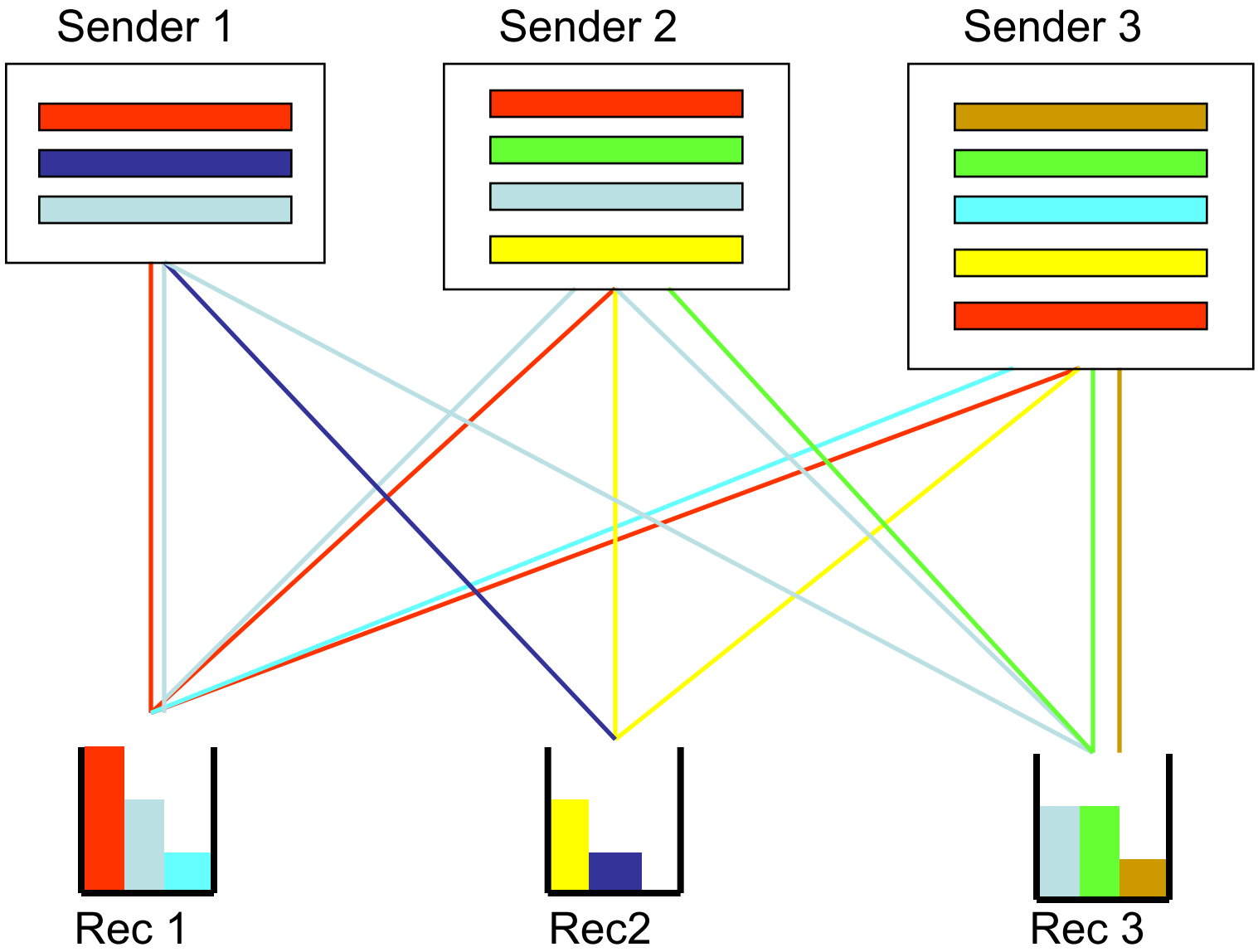


Applications: Path Diversity



algolma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

Applications: Peer-2-Peer



algolma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

Fountain Codes

Sender sends a potentially limitless stream of encoded bits.

Receivers collect bits until they are reasonably sure that they can recover the content from the received bits, and send STOP feedback to sender.

Automatic adaptation: Receivers with larger loss rate need longer to receive the required information.

Want that each receiver is able to recover from the minimum possible amount of received data, and do this efficiently.

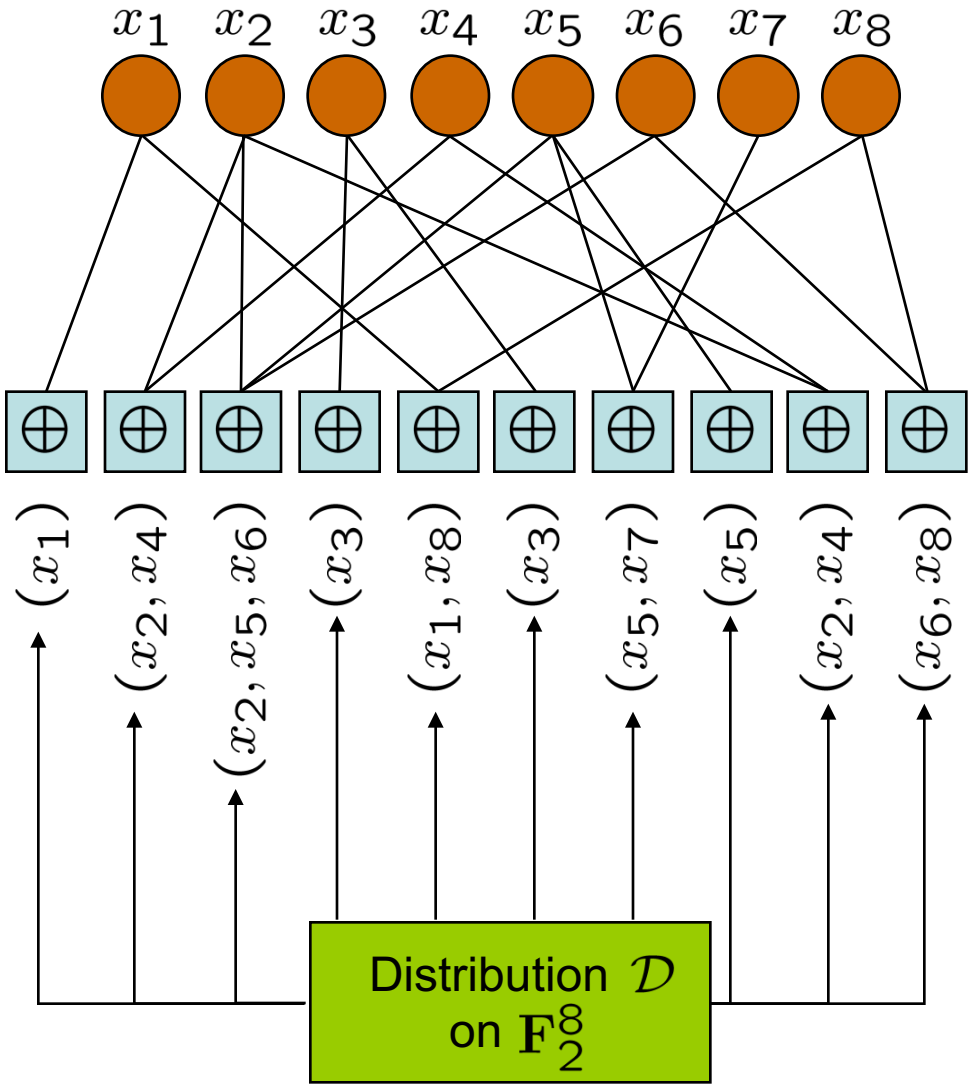
Fountain Codes

Fix distribution $\mathcal{D}$ on $\mathbf{F}_2^k$, where k is number of input symbols.

For every output symbol sample independently from $\mathcal{D}$ and add input symbols corresponding to sampled subset.

algolma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

Fountain Codes



algolma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

Universality and Efficiency

[Universality]
Want sequences of Fountain Codes for which the overhead is arbitrarily small

[Efficiency]
Want per-symbol-encoding to run in close to constant time, and decoding to run in time linear in number of output symbols.

LT-Codes

- Invented by Michael Luby in 1998.
- First class of universal and almost efficient Fountain Codes
- Output distribution has a very simple form
- Encoding and decoding are very simple

LT-Codes

LT-codes use a restricted distribution on $\mathbf{F}_2^k$:

Fix distribution $\{\Omega_1, \Omega_2, \ldots, \Omega_k\}$ on $\{1, 2, \ldots, k\}$

Distribution $\mathcal{D}$ is given by

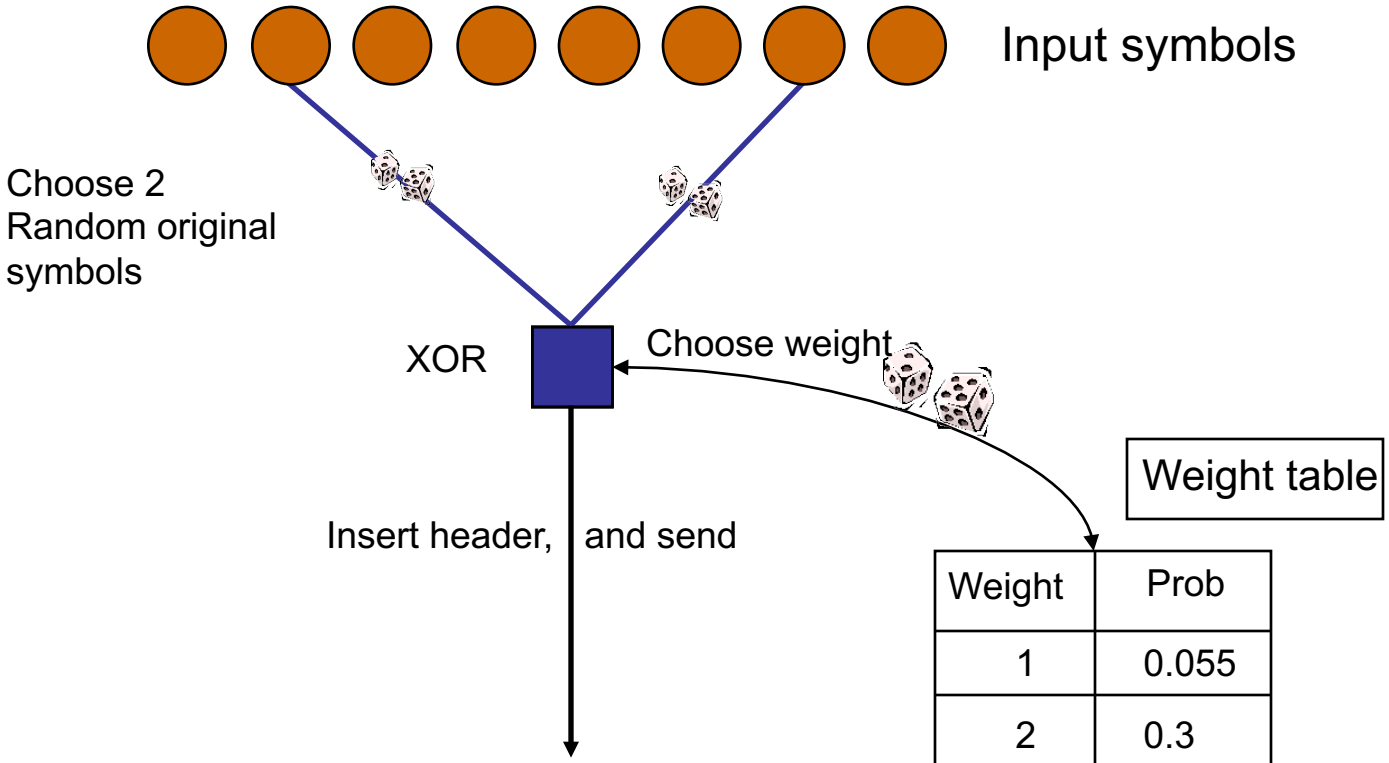
$$\mathsf{Prob}_{\mathcal{D}}(x) = \frac{\Omega_w}{\binom{k}{w}}$$

where w is the Hamming weight of x

Parameters of the code are $(k, \Omega(x))$

$$\Omega(x) = \Omega_1 x + \Omega_2 x^2 + \cdots + \Omega_k x^k$$

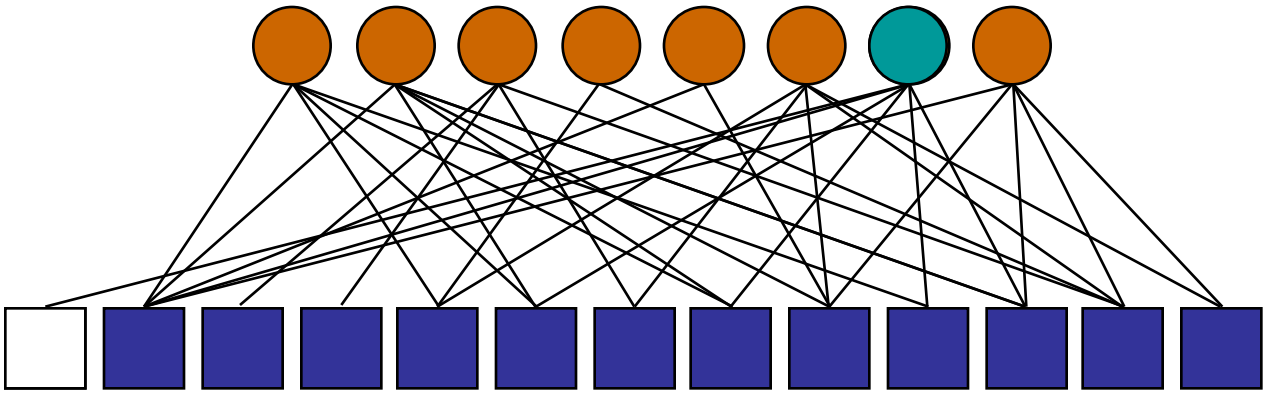
The LT Coding Process



Weight	Prob
1	0.055
2	0.3
3	0.1
4	0.08
⋮	⋮
100000	0.0004

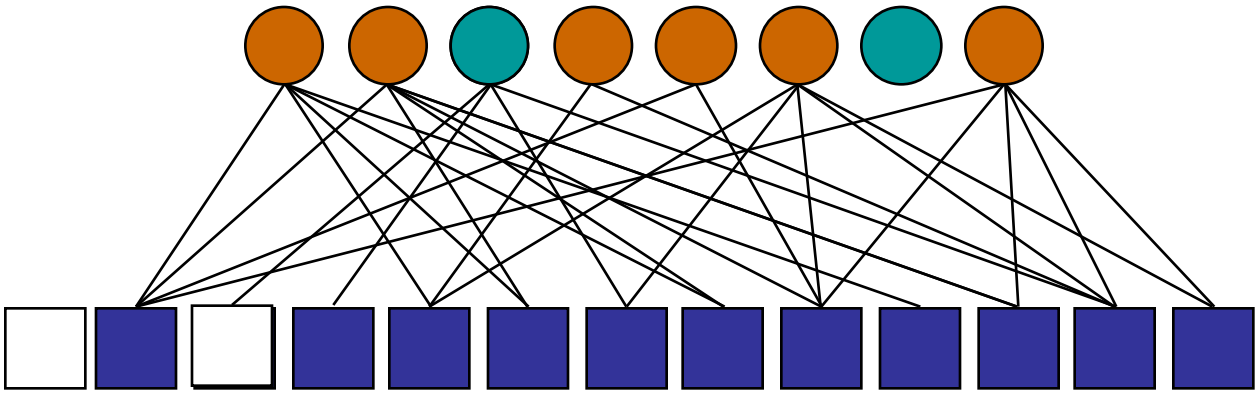
algo lma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

Decoding



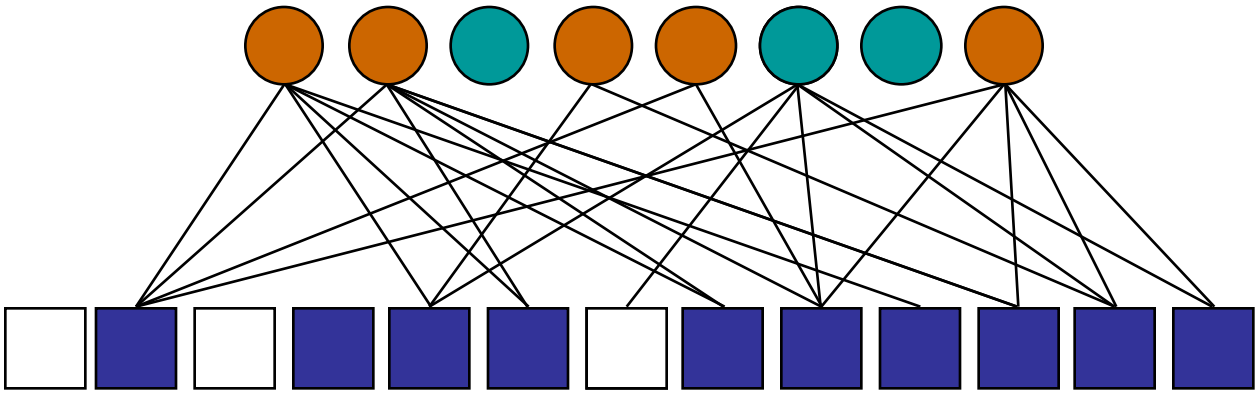
algolma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

Decoding



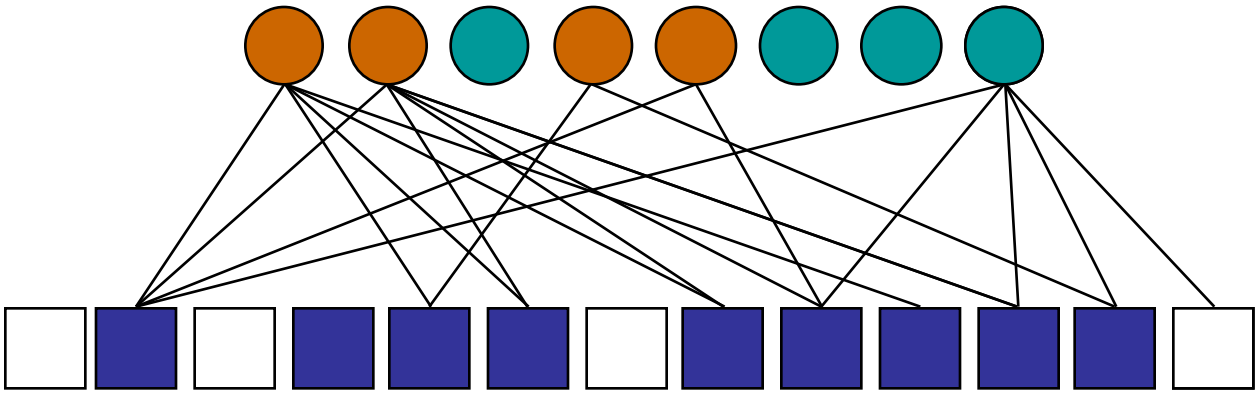
algolma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

Decoding



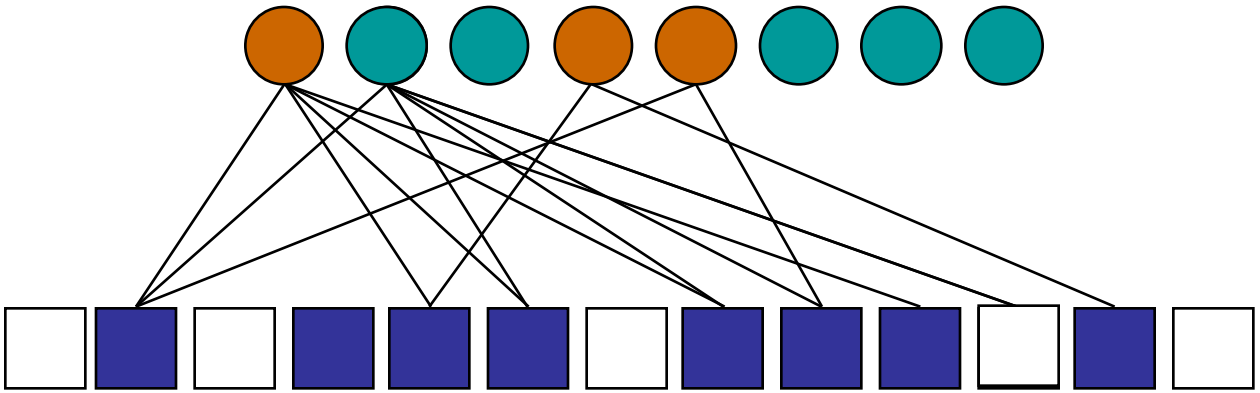
algo lma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

Decoding



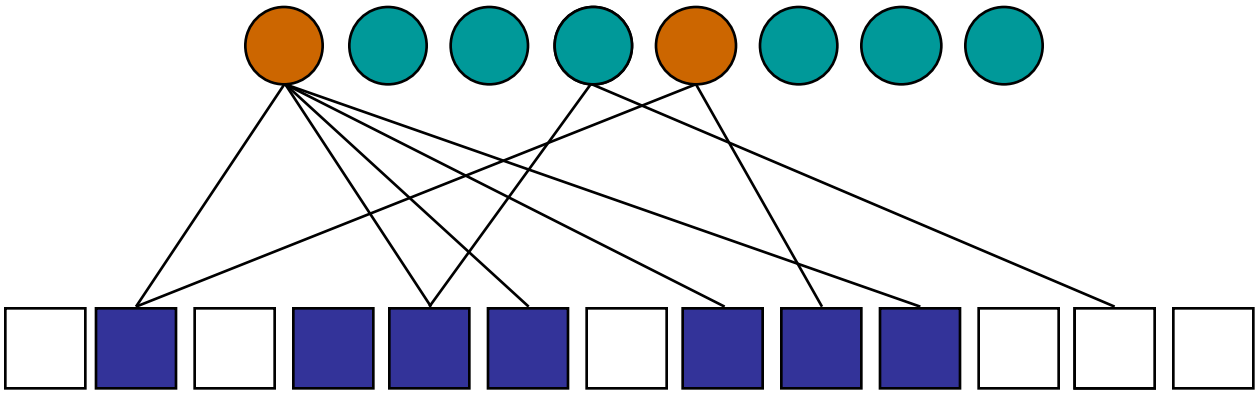
algo lma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

Decoding



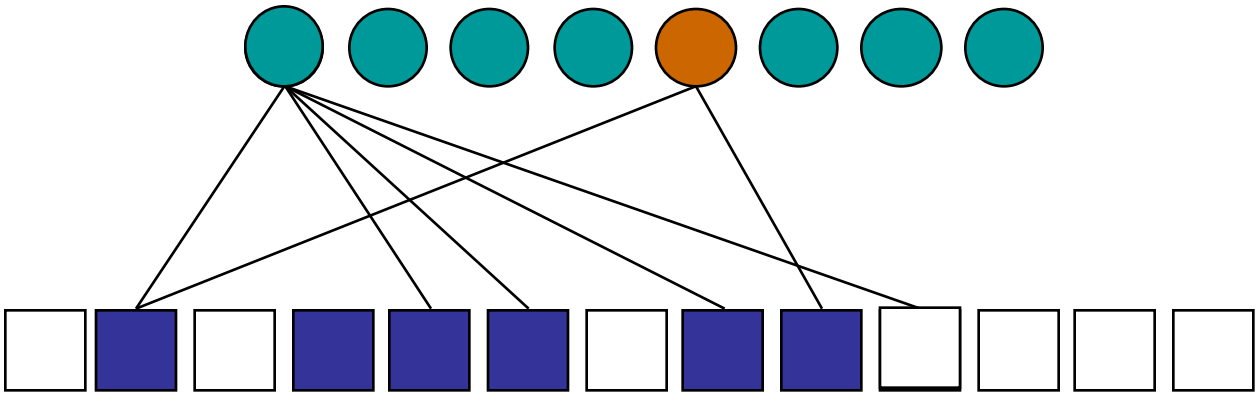
algo⊕lma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

Decoding



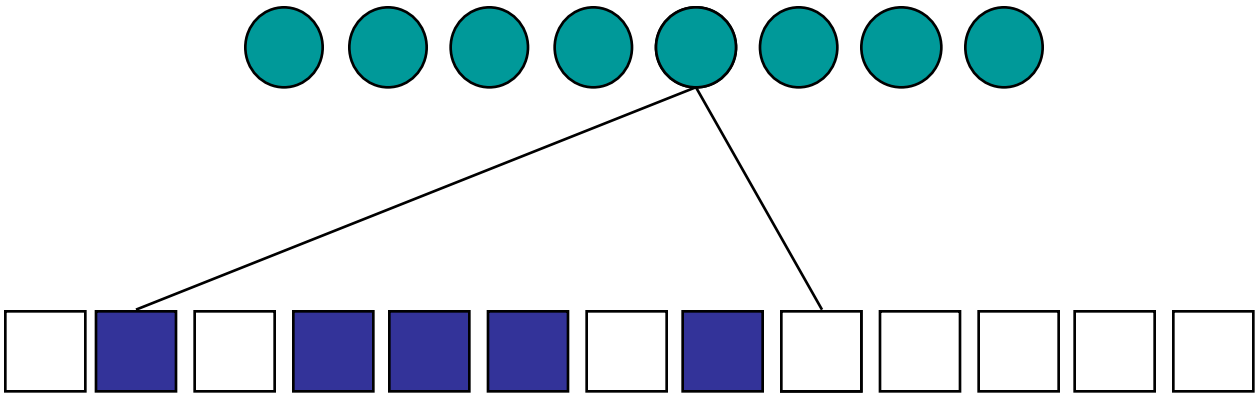
algolma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

Decoding



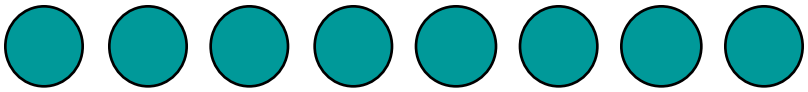
algo lma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

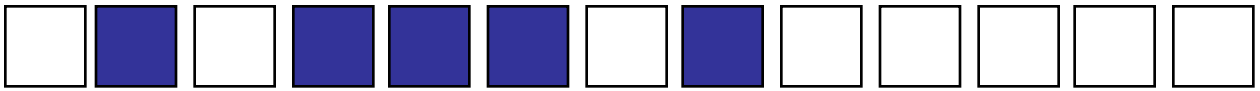
Decoding



algolma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

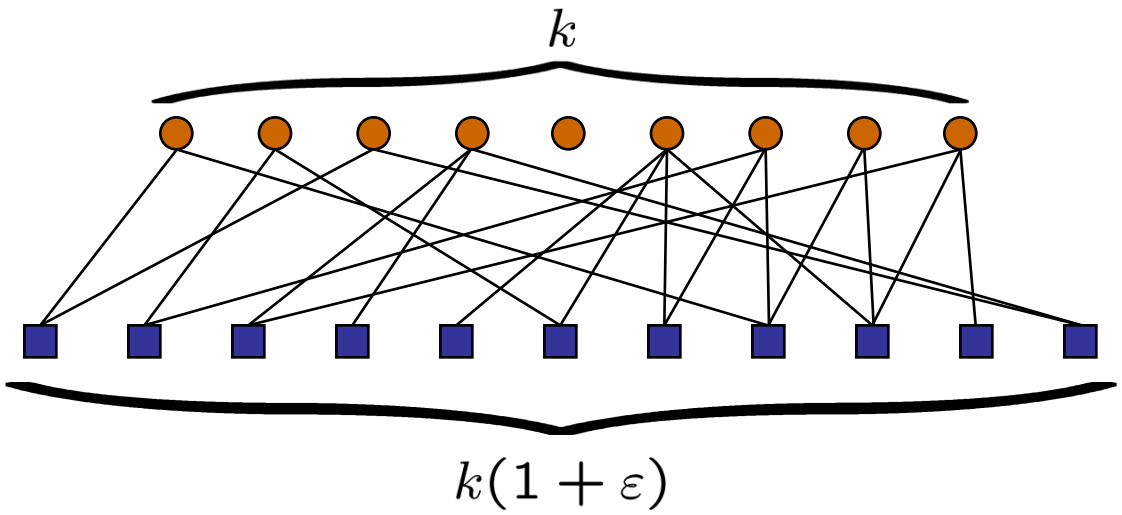
Decoding





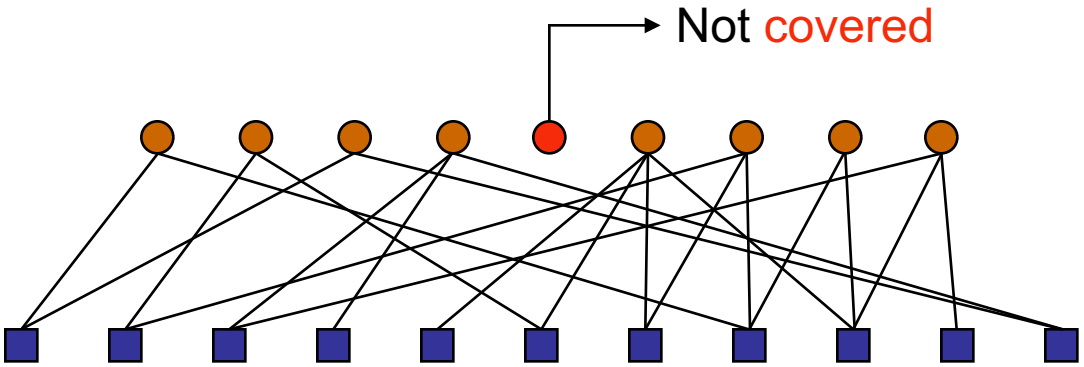
algolma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

Average Degree of Distribution should be $O(\log(k))$



algolma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

Average Degree of Distribution should be $O(\log(k))$

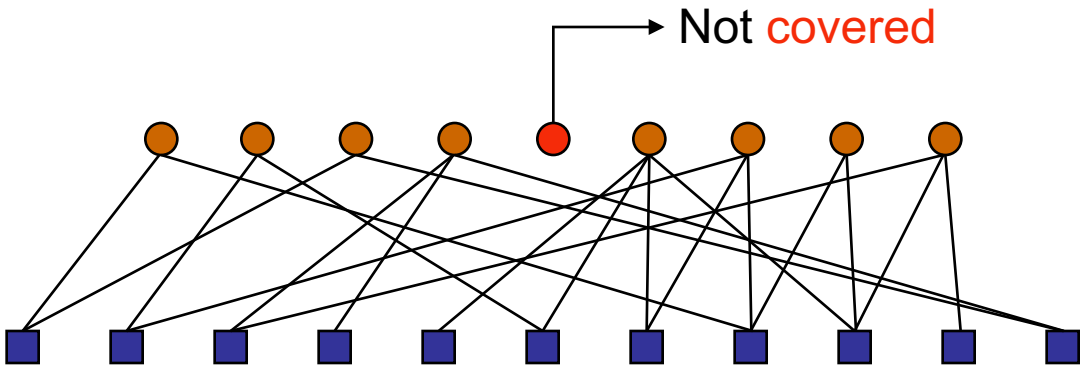


Prob. Decoding error $\geq$ Prob. Non-coverage

$$\geq \left(1 - \frac{1}{k}\right)^{k(1+\varepsilon)\Omega'(1)}$$

$$\simeq e^{-(1+\varepsilon)\Omega'(1)}$$

Average Degree of Distribution should be $O(\log(k))$



$$e^{-(1+\varepsilon)\Omega'(1)} \leq \frac{1}{k} \Leftrightarrow \Omega'(1) \geq \frac{\ln(k)}{1+\varepsilon}$$

Luby has designed universal LT-codes with average degree around $O(\log(k))$ and overhead $O(\log^2(k)/\sqrt{k})$

So:

Average degree constant means error probability constant

How can we achieve constant workload per output symbol, and still guarantee vanishing error probability?

Raptor codes achieve this!

algolma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

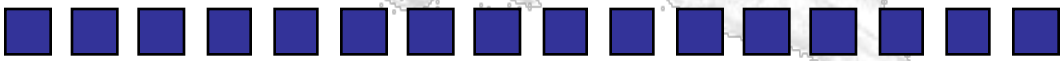
Raptor Codes

Input symbols

Traditional pre-code

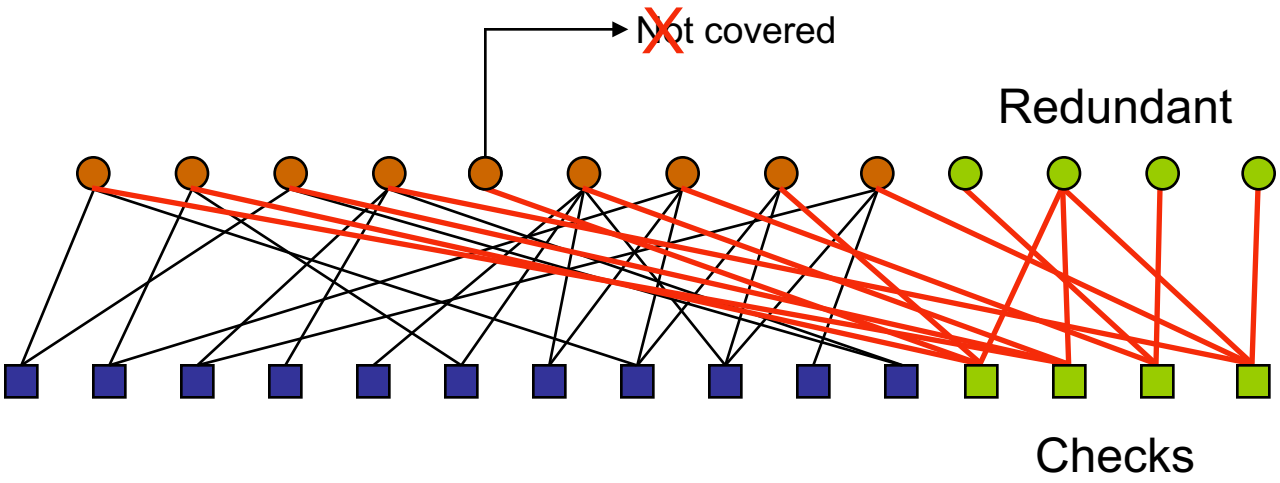
δ – fraction erasures

LT-light



Output symbols

Raptor Codes



If pre-code is chosen properly, then the LT-distribution can have constant average degree, leading to linear time encoding.

Raptor Code is specified by the input length k, precode $\mathcal{C}$ and output distribution $\Omega(x)$.

How do we choose $\Omega(x)$ and $\mathcal{C}$?

algo lma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

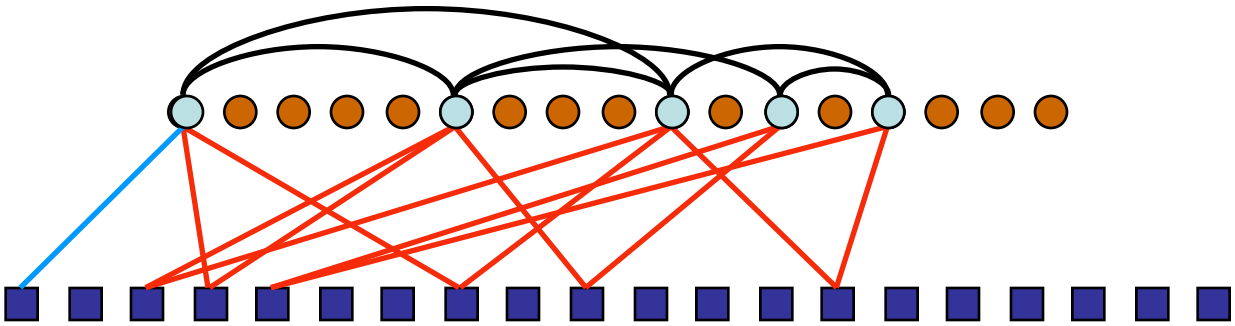
Special Raptor Codes: LT-Codes

LT-Codes are Raptor Codes with trivial pre-code: Need average degree $O(\log(k))$

LT-Codes compensate for the lack of the pre-code with a rather intricate output distribution.

Progressive Giant Component Analysis

A different method for the analysis of the decoder:



Want enough nodes of degree 2 so there exists a giant component in the induced (random) graph on input symbols.

Progressive Giant Component Analysis

First giant component removes α-fraction of input symbols.

Residual distribution: $\Omega(x(1-\alpha)+\alpha)$

Fraction of residual nodes of degree 2: $(1-\alpha)^2\Omega''(\alpha)$

Average degree of new induced graph: $(1-\alpha)\Omega''(\alpha)$

Condition: $(1-\alpha)\Omega''(\alpha)=1$

"Ideal distribution:" $\Omega(\alpha)=\frac{\alpha^2}{1\cdot 2}+\frac{\alpha^3}{2\cdot 3}+\cdots$

algolma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

Progressive Giant Component Analysis

Analysis does not use “tree-assumption”, but only properties of induced graph.

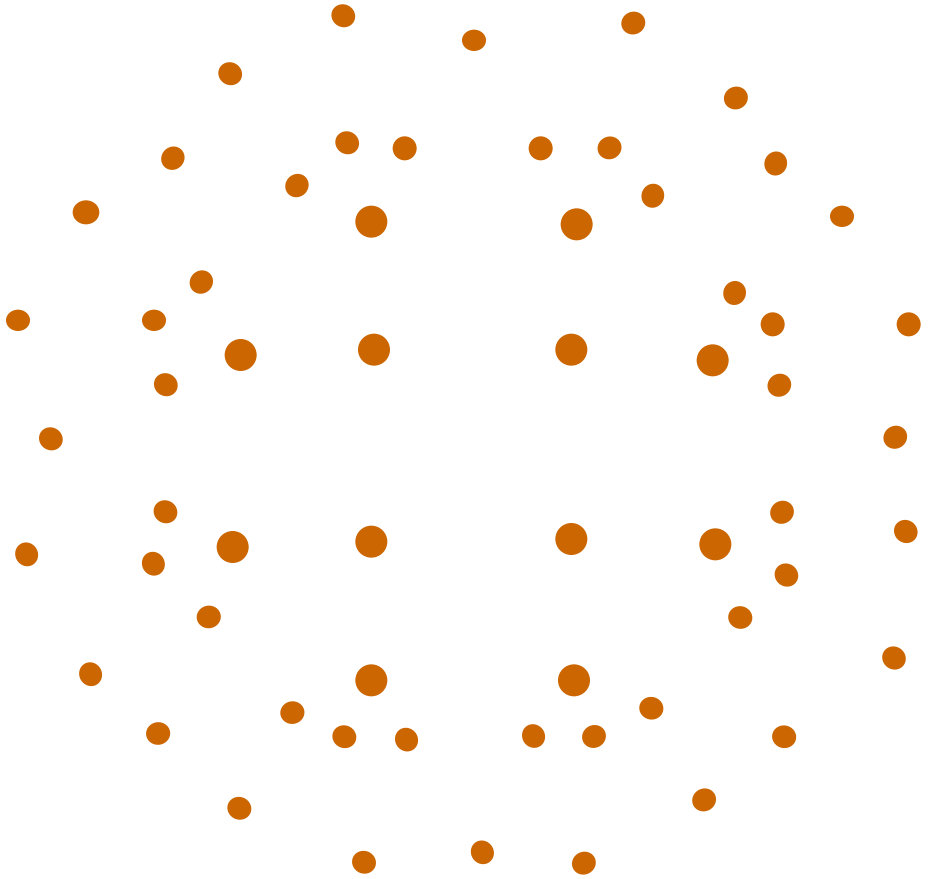
Analysis can be used to obtain error bounds for the decoding algorithm.

It can also be used to obtain capacity-achieving distributions on the erasure channel.

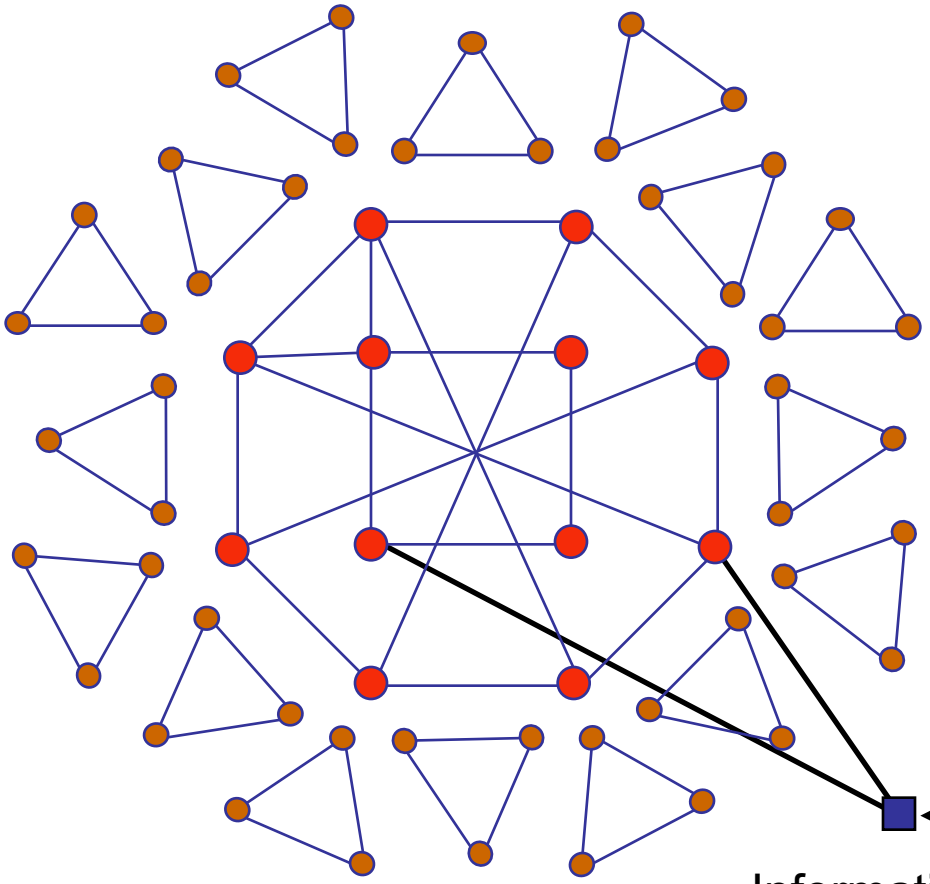
A modified version can be used to obtain Ω_2 for capacity-achieving distributions for other symmetric channels.

algolma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

Nodes of Degree 2



Nodes of Degree 2



New output node of degree 2

Information Loss!

Fraction of Nodes of Degree 2

If there exists component of linear size (i.e., a giant component), then next output node of degree 2 has constant probability of being useless.

Therefore, graph should not have giant component.

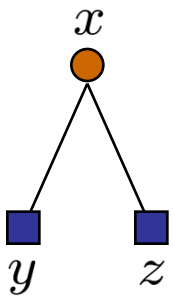
This means that for capacity achieving degree distributions we must have: $\Omega_2 \leq \frac{1}{2}$.

On the other hand, if $\Omega_2 < \frac{1}{2}$ then algorithm cannot start successfully.

So, $\Omega_2 = \frac{1}{2}$ for capacity-achieving codes:

The q-ary symmetric channel (large q)

Double verification decoding (Luby-Mitzenmacher):



If y and z are correct, then they verify x. Remove all of them from graph and continue.

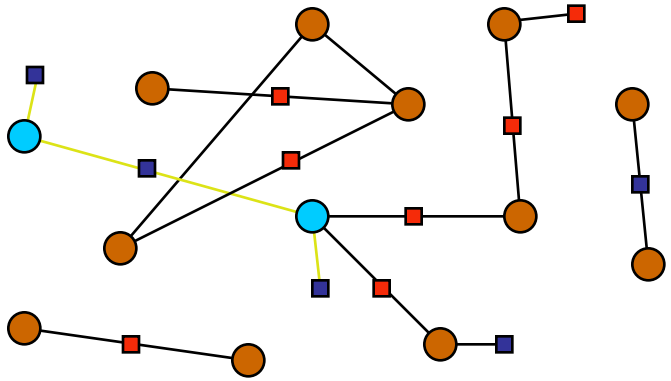
Can be shown that number of correct output symbols needs to be at least

$$3 + \frac{1}{\mathrm{e}} - \frac{\mathrm{e}}{2} \simeq 2.00873$$

Times number of input symbols.

The q-ary symmetric channel (large q)

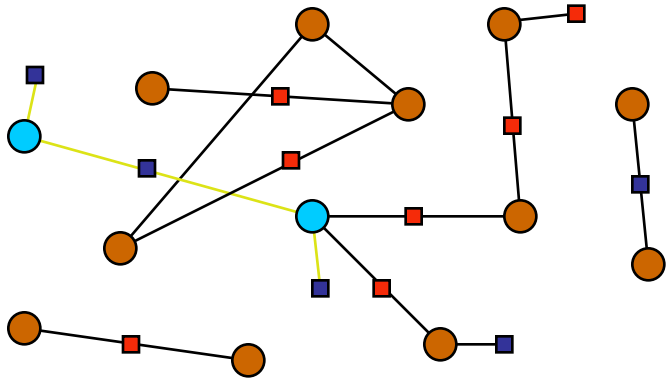
More sophisticated algorithms: induced graph!



If two input symbols are connected by a correct output symbol, and each of them is connected to a correct output symbol of degree one, then the input symbols are verified. Remove from them from graph.

The q-ary symmetric channel (large q)

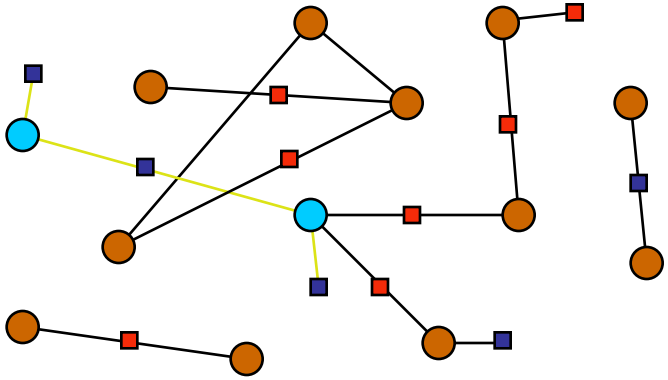
More sophisticated algorithms: induced graph!



More generally: if there is a path consisting of correct edges, and the two terminal nodes are connected to correct output symbols of degree one, then the input symbols get verified. (More complex algorithms.)

The q-ary symmetric channel (large q)

Limiting case: Giant component consisting of correct edges, two correct output symbols of degree one “poke” the component. So, ideal distribution “achieves” capacity.



Binary Memoryless Symmetric Channels

What is the fraction of nodes of degree 2 for capacity-achieving Raptor Codes?

$$\Omega_2(\mathcal{C}) := \frac{1}{2}\Pi(\mathcal{C})$$

where, in general

$$\Pi(\mathcal{C}) := \frac{\mathsf{Cap}(\mathcal{C})}{\mathsf{E}[\tanh(Z/2)]}$$

and Z is the LLR of the channel.

$\Omega_2(\mathcal{C})$

$$\Omega_2(\mathsf{BEC}(p)) = \frac{1}{2}$$

$$\Omega_2(\mathsf{BSC}(p)) = \frac{1-h(p)}{2(1-2p)^2}$$

$$\Omega_2(\mathsf{BIAWGN}(\sigma)) = \frac{1}{4\ln(2)}(1+O(1/\sigma)).$$

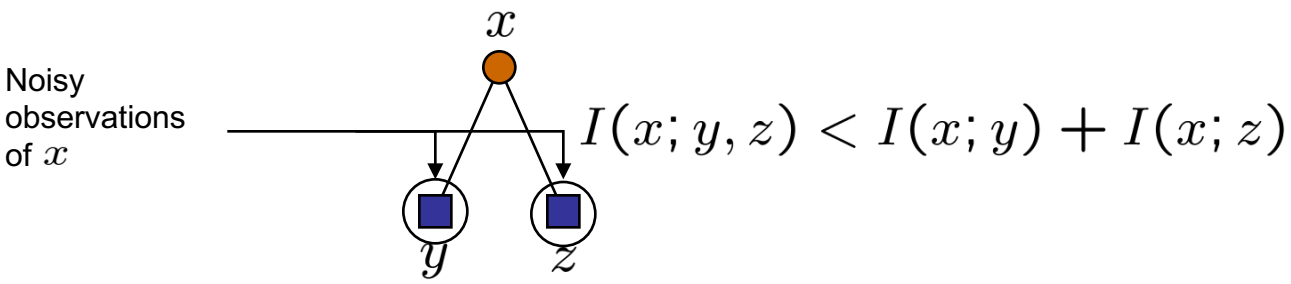
General Symmetric Channels: Mimic Proof

Proof is information theoretic: if fraction of nodes of degree 2 is larger by a constant, then :

- Expectation of the hyperbolic tangent of messages passed from input to output symbols at given round of BP is larger than a constant.
- This shows that $I(x; z_2) < n\Omega_2(\mathsf{Cap}(\mathcal{C}) - \tau)$
- So code cannot achieve capacity.

General Symmetric Channels: Mimic Proof

Fraction of nodes of degree one for capacity-achieving Raptor Codes:



Therefore, if $\Omega_1 > 0$, and if $z_1, \ldots, z_m$ denote output nodes of degree one, then

$$I(x; z_1, \ldots, z_m) < \sum_{i=1}^{m} \mathsf{Cap}(\mathcal{C}) - \eta$$

So $\Omega_1 = 0$

Very Good Degree Distribution

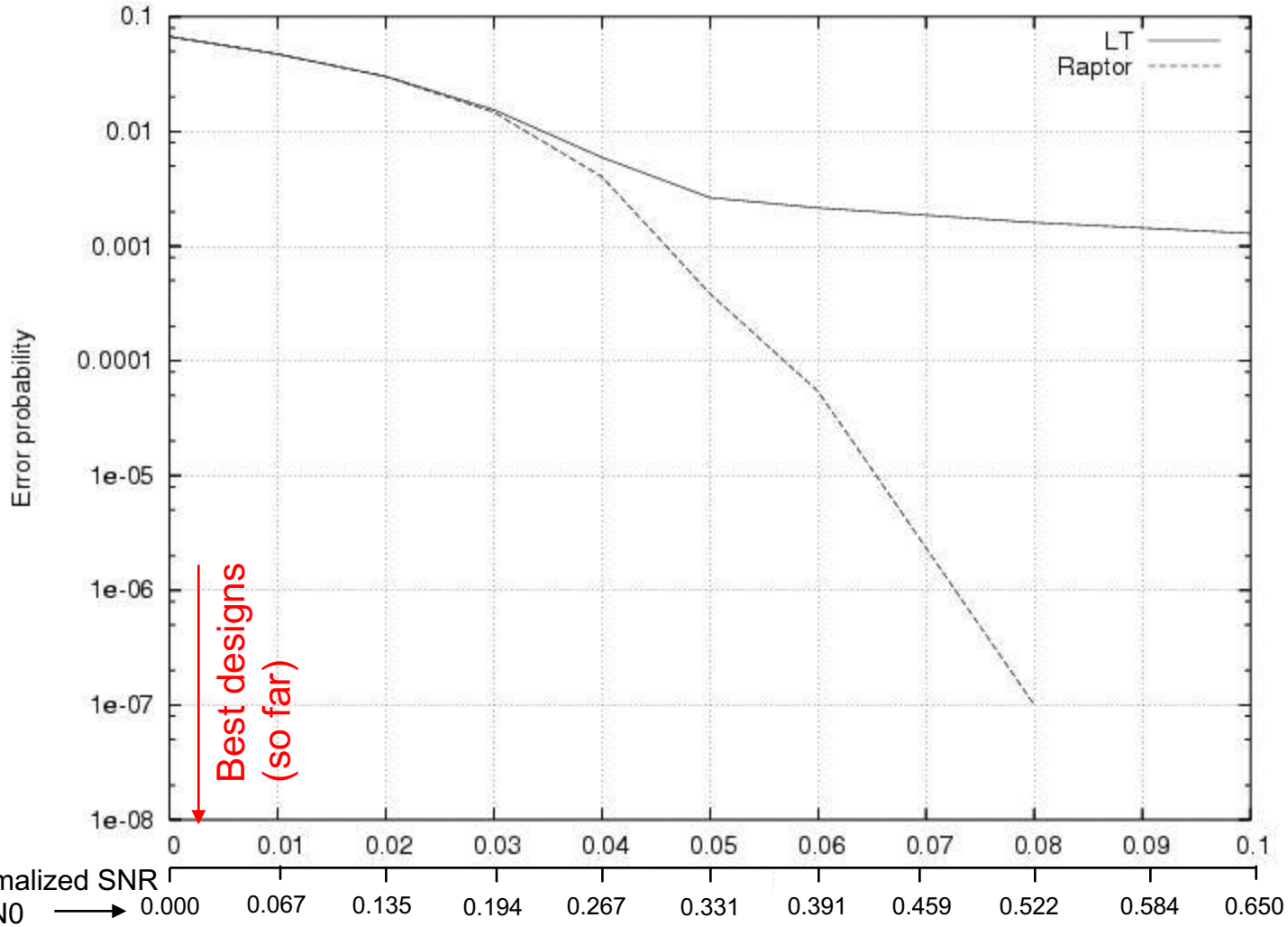
$$\Omega_{\mathcal{C}}(x) = \frac{1 - (1-x)^{\Pi(\mathcal{C})} - \Pi(\mathcal{C})}{1 - \Pi(\mathcal{C})}.$$

In the case of the BEC this distribution is equal to the ideal distribution (hence generalization).

Uses certain threshold phenomena in random graphs.

algo lma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

Sequences Designed for the BEC



$\sigma = 0.978$

Conclusions

- For LT- and Raptor codes, some decoding algorithms can be phrased directly in terms of subgraphs of graphs induced by output symbols of degree 2.
- This leads to a simpler analysis without the use the tree assumption.
- For the BEC, and for the q-ary symmetric channel (large q) we obtain essentially the same limiting capacity-achieving degree distribution, using the giant component analysis.
- An information theoretic analysis gives the optimal fraction of output nodes of degree 2 for general memoryless symmetric channels.
- A graph analysis reveals very good degree distributions, which perform very well experimentally.