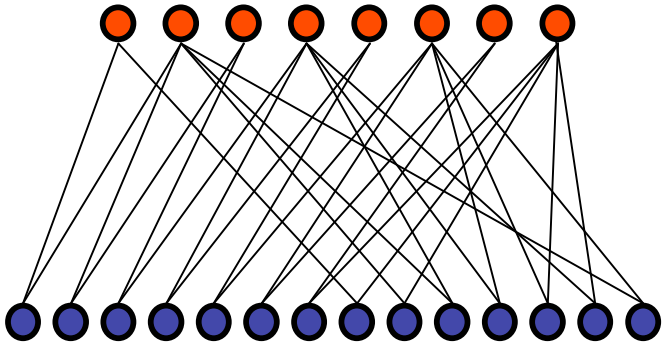
algolma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

Verification Decoding of Raptor Codes



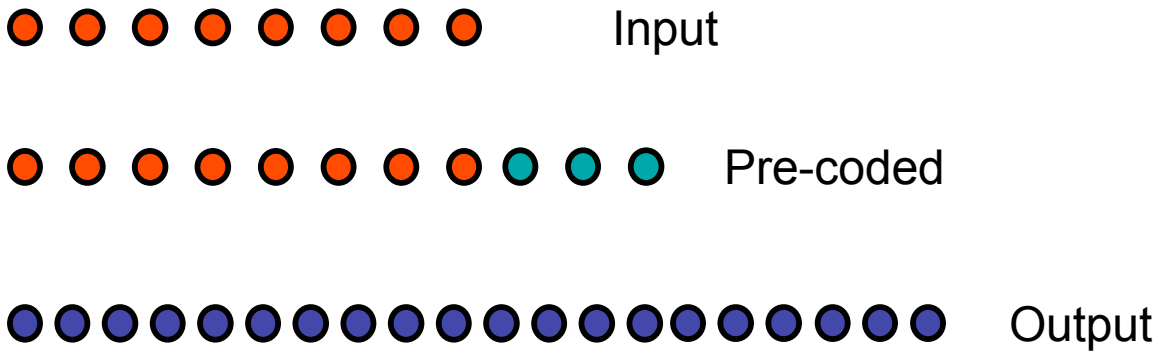
Amin Shokrollahi
Laboratoire d'algorithmique
Laboratoire de mathematiques algorithmiques
EPFL

Joint work with R. Karp and M. Luby

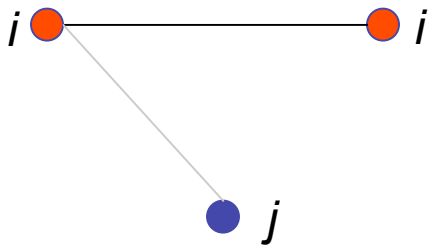
algolma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

Raptor Codes

Considering Raptor Codes of type $(k, \Omega(x), \mathtt{C})$: k is the number of input symbols, $\Omega(x)$ the degree distribution, and $\mathtt{C}$ is the pre-code.



q-ary Symmetric Channel



Capacity $\sim 1 - p$

Want to code on this channel and get arbitrarily close to the capacity.

Possible Solutions

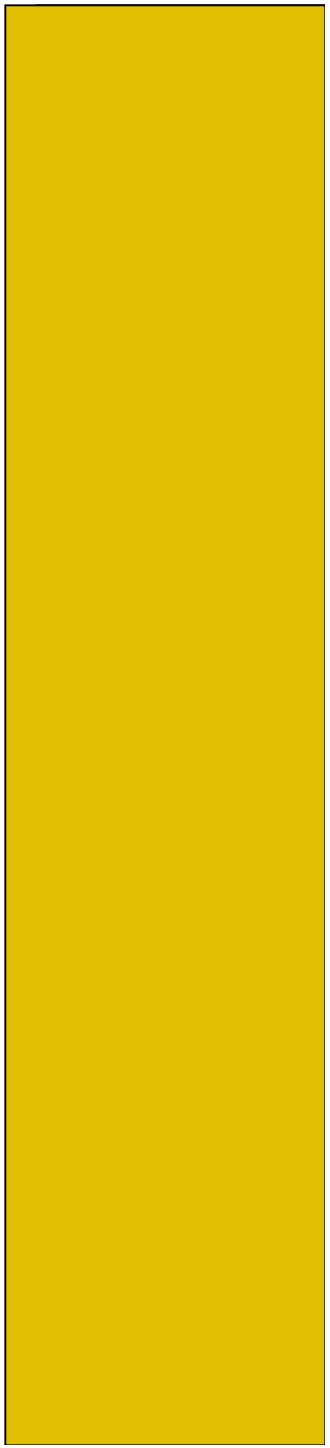
Can use a part of every symbol as a hash, detect symbols for which hash doesn't match, and reduce to erasure coding.

However, error probability would not be very good.

Will use adaptation of "verification decoding" instead.

Method was introduced by Luby and Mitzenmacher for decoding of LDPC codes on the q-ary symmetric channel.

We will assume that q is very large in the following. The error probabilities will always be at least of the order of 1/q.



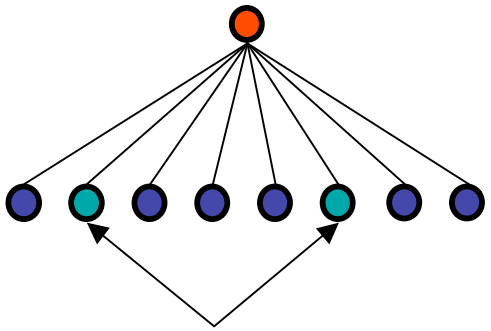
algolma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

Simple Algorithm

For every input symbol if two neighbors are of degree one and are equal, decode the input symbol to that common value.

Add value of input symbol to neighbors, and remove input symbol from graph.

Continue.



Verify input symbol

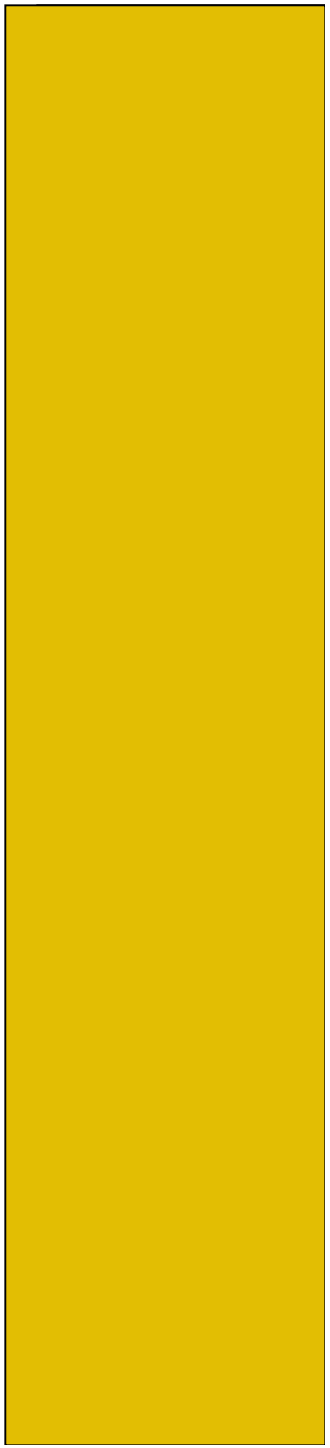
What is the Best we can Expect?

Every input symbol needs two correctly transmitted output symbols for recovery.

So, number of correctly transmitted output symbols has to be at least twice the number of input symbols.

Capacity result says that we should be able to correct if number of correctly transmitted symbols is roughly equal to number of input symbols.

So, we can correct only at most to half of the “capacity”. But can we achieve this?



algolma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

Message Passing Formulation

Input symbol v
Output symbol c

Message space: $\mathrm{GF}(q) \cup \{\mathtt{E}\}$

$$\mathtt{m}_{cv} = \begin{cases} \mathtt{E}, & \text{if } \exists\, v' \neq v\colon\ \mathtt{m}'_{v'c} = \mathtt{E}, \\ \displaystyle\sum_{v' \neq v} \mathtt{m}'_{v'c}, & \text{else,} \end{cases}$$

$$\mathtt{m}'_{vc} = \begin{cases} m, & \text{if } \exists\, c' \neq c'', c', c'' \neq c\colon \\ & \mathtt{m}_{c'v} = \mathtt{m}_{c''v} = m, \\ \mathtt{E}, & \text{else.} \end{cases}$$

algolma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

Message Passing Formulation

$$p_i = \Pr[\mathtt{m}_{vc}^{(i)} = \mathtt{E}]$$

α Average degree of input symbols

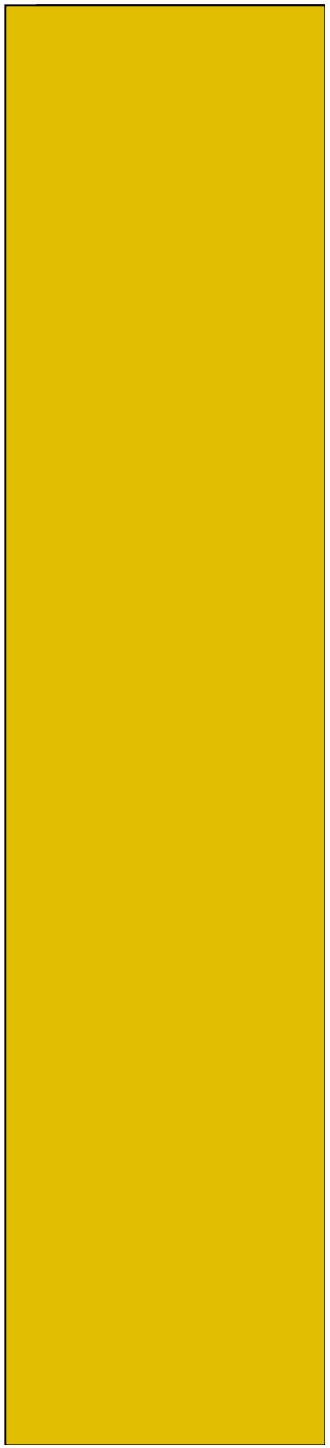
$$\omega(x) = \frac{\Omega'(x)}{\Omega'(1)}$$

$$p_{i+1} = \mathrm{e}^{-\alpha\omega(1-p_i)}(1 + \alpha\omega(1 - p_i)$$

Want:

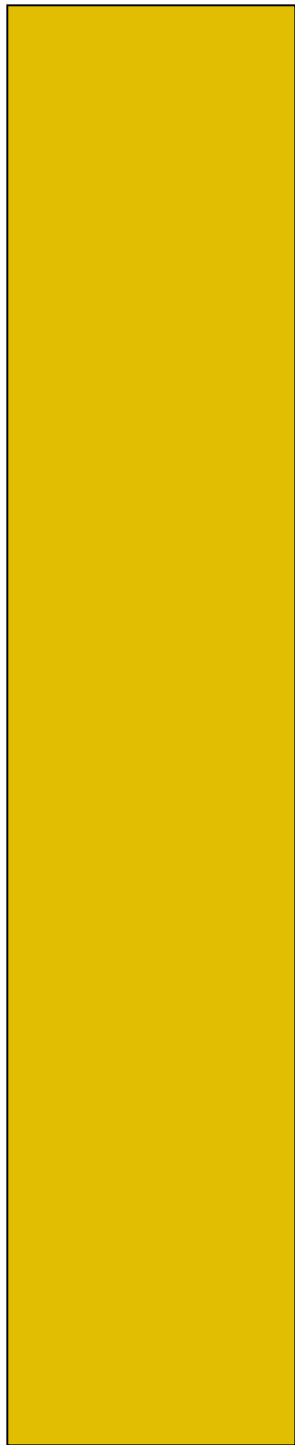
$$x < 1 - \mathrm{e}^{-\alpha\omega(x)}(1 + \alpha\omega(x))$$

for $0 < x < 1 - \epsilon$.



algolma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

A Little Theorem



algo lma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

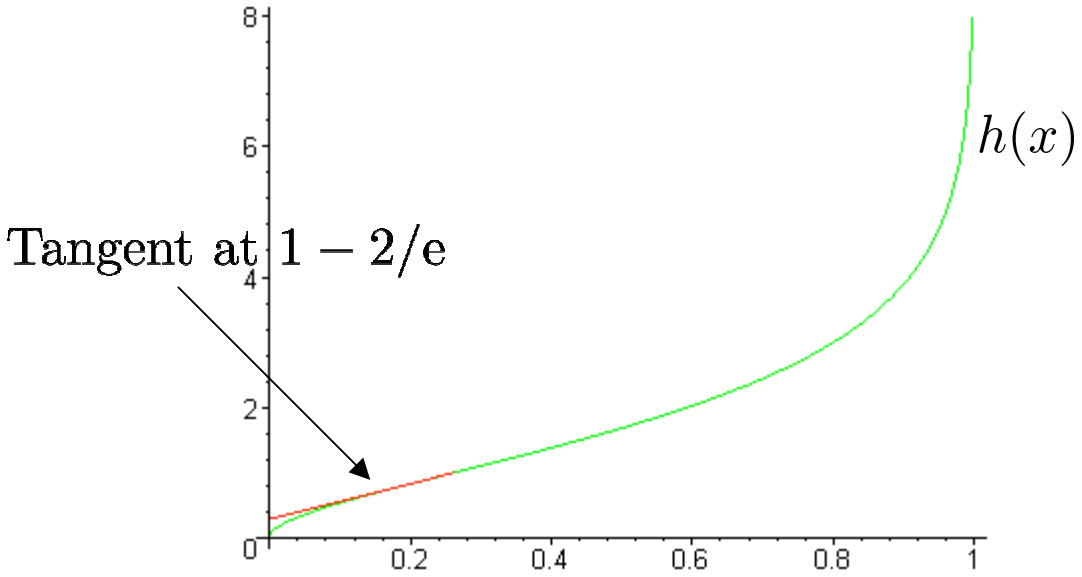
And its Corollary

Overhead of the algorithm: How much more output symbols are needed than the minimum necessary as a fraction of the number of input symbols.

Overhead = $\int_0^1 \alpha\omega(x)\mathrm{d}x - 1$

The overhead of the simple algorithm is at least $2+1/\mathrm{e}-\mathrm{e}/2-f(\epsilon) \simeq 1.00873-f(\epsilon)$.

(Sketch of the) Proof of the Theorem



Raptor Code

This would take care of the LT-code. What about the Raptor code?

At the end of the LT-decoding the input symbols that are not yet determined are regarded as erasures.

Pre-code needs to clean up erasures only. Total overhead:

$$\frac{1}{x_0}\left(3+\frac{1}{\mathrm{e}}-\frac{\mathrm{e}}{2}-\int_{h(x_0)}^{\infty} y^2\mathrm{e}^{-y}\mathrm{d}y+(1-x_0)h(x_0)\right)-1$$

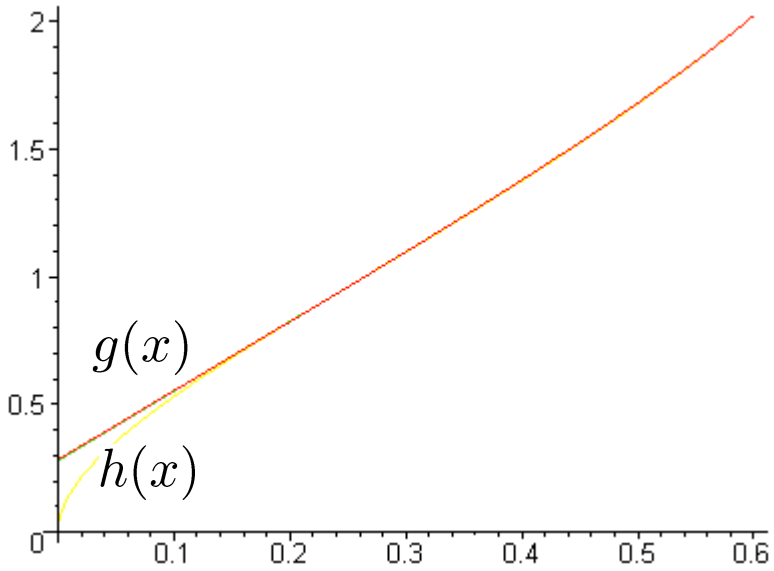
where

$$x_0 = 1-\epsilon$$

algo lma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

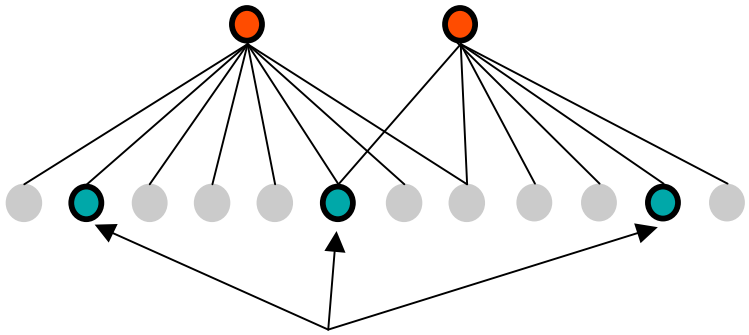
Good Degree Distributions

gives overhead of 2.044606.



algolma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

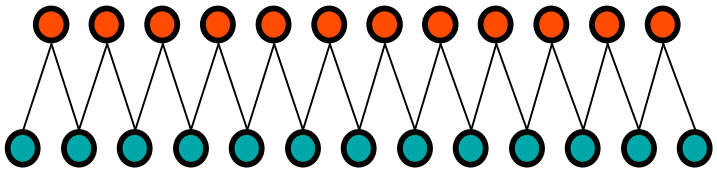
Better Algorithms



Recover and verify the two input symbols

Need three correctly transmitted output symbols for every two input symbols, so best overhead would be 3/2.

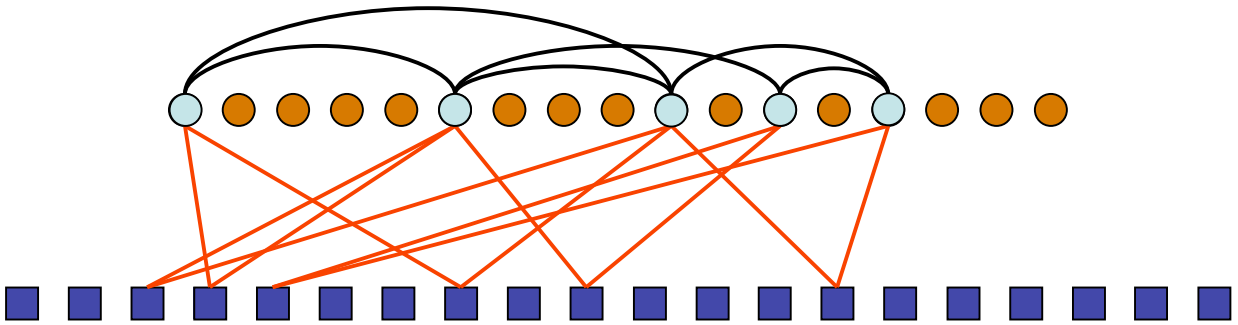
Better Algorithms



As the length of the paths grow, we expect to become more efficient in terms of the overhead, but the algorithms become more complicated.

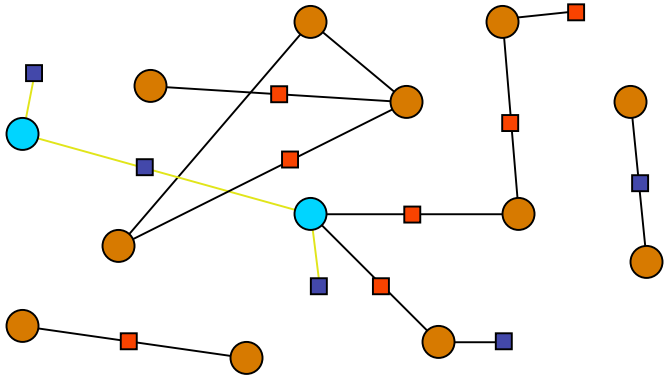
algolma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

Induced Graph



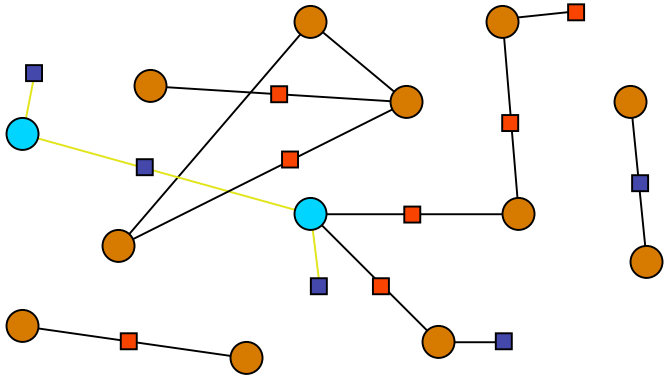
algolma
laboratoire d'algorithmique
laboratoire de mathematiques algorithmique

Decoding on the 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.

Asymptotic Case



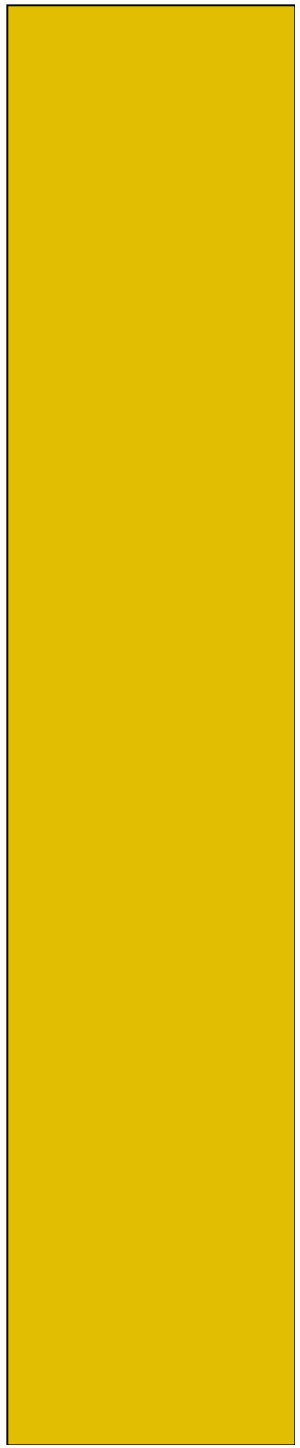
If the graph formed by the correctly transmitted edges in the induced graph has a giant component, and the component is “poked” by two correctly transmitted output symbols of degree one, then the corresponding input symbols are verified and can be removed from the graph.

Asymptotic Case

Similar to the case of the erasure channel.

The Soliton distribution comes arbitrarily close to the capacity, and a low-rate pre-code can clean up the residual errors.

Algorithms become exceedingly complicated, though. (Finding the paths generated by correct edges can be difficult when paths are long.)



Conclusions

Verification decoding is a simple algorithm that can be used for decoding on the q-ary symmetric channel, for large q.

Simple algorithm can be easily analyzed and good codes can be designed, regardless of the error probability of the q-ary symmetric channel.

A class of algorithms can be designed that are of increasing complexity, but lead to codes of smaller overhead.

Analysis can be reduced to the giant components of the induced graphs.