Insertion of erasures as a method of q-ry LDPC codes decoding

Alexey Frolov, Victor Zyablov

Inst. for Information Transmission Problems
Russian Academy of Sciences
Email: {alexey.frolov, zyablov}@iitp.ru

September 8, 2010

Outline

Task statement

To develop a decoding algorithm for q-ry LDPC codes capable of correcting both errors and erasures and research its realized correcting capabilities.

Gallager's LDPC codes

Parity-check matrix of Gallager's LDPC-code

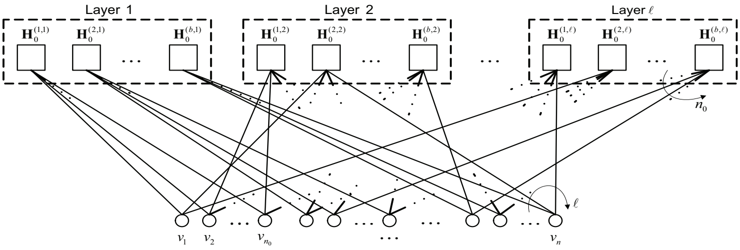
$$\mathbf{H} = \begin{pmatrix} \pi_1(\mathbf{H}_b) \\ \pi_2(\mathbf{H}_b) \\ \vdots \\ \pi_\ell(\mathbf{H}_b) \end{pmatrix}_{\ell b \times b n_0}$$

where

$$\mathbf{H}_b = \begin{pmatrix} 11\ldots1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & 11\ldots1 & \ldots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \ldots & 11\ldots1 \end{pmatrix}_{b \times b n_0}$$

1. $(n_0, n_0 - 1)$ single parity-check (SPC) codes are constituent codes
2. ℓ random column permutations of $\mathbf{H}_b$ form layers of $\mathbf{H}$
3. Code rate is $R \geqslant 1 - \frac{\ell b}{b n_0}$

Bipartite Tanner graph of LDPC codes



1. Constraint nodes have degree n_0 and represent constituent codes.
2. Variable nodes have degree ℓ and represent codesymbols. Each variable node is connected to exactly one constraint node in each layer.

Generalized LDPC codes

Parity-check matrix of generalized LDPC-code

$$\mathbf{H} = \begin{pmatrix} \pi_1(\mathbf{H}_b) \\ \pi_2(\mathbf{H}_b) \\ \vdots \\ \pi_\ell(\mathbf{H}_b) \end{pmatrix}_{\ell b(n_0-k_0)\times bn_0}$$

where

$$\mathbf{H}_b = \begin{pmatrix} \mathbf{H}_0 & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_0 & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{H}_0 \end{pmatrix}_{b(n_0-k_0)\times bn_0}$$

1. (n_0, k_0) codes are constituent codes
2. Code rate is $R \geqslant 1 - \frac{\ell b(n_0-k_0)}{bn_0} \Rightarrow \frac{k_0}{n_0} > 1 - \frac{1}{\ell}$

q-ry LDPC code obtained from generalized LDPC code structure

Constituent code parity-check matrix:

$$\mathbf{H}_0 = \underbrace{\begin{pmatrix} 1 & \alpha & \ldots & \alpha^{n_0-1} \end{pmatrix}}_{n_0}, \alpha \in GF(q) \setminus \{0\}$$

so

$$\mathbf{H}_b = \begin{pmatrix} 1\ \alpha \ldots\ \alpha^{n_0-1} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & 1\ \alpha \ldots\ \alpha^{n_0-1} & \ldots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \ldots & 1\ \alpha \ldots\ \alpha^{n_0-1} \end{pmatrix}_{b \times bn_0}$$

Generalized syndrome

Received vector is $\mathbf{r} = \mathbf{v} + \mathbf{e}$. The syndrome vector is the ℓbm-tuple:

$$\mathbf{S} = \mathbf{r}\mathbf{H}^T$$

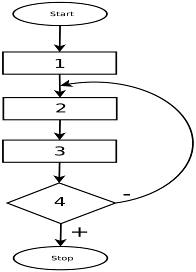
The **generalized syndrome** is the ℓb-tuple:

$$\mathbf{S} = (\mathbf{s}_1\mathbf{s}_2 \ldots \mathbf{s}_\ell) = (\mathbf{s}_{1,1}\mathbf{s}_{2,1} \ldots \mathbf{s}_{b,1} \ldots \mathbf{s}_{1,\ell}\mathbf{s}_{2,l} \ldots \mathbf{s}_{b,\ell})$$

where $\mathbf{s}_{i,j}$ is an m-tuple.

Decoding algorithm block-scheme

The decoding algorithm is required to correct both errors and erasures. In each iteration we will replace error-suspicious symbols with erasures and then perform only the erasure correcting within the iteration.

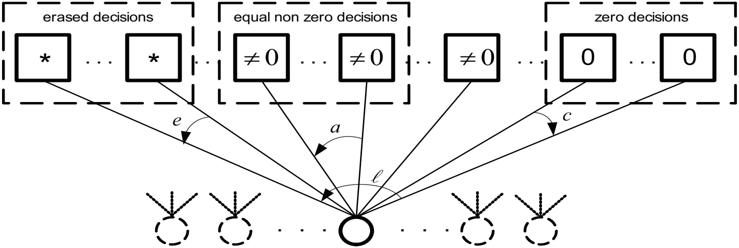


1. Initialization
2. Error-suspicious symbols erasing
3. Correcting of erasures
4. Stop criteria

Initialization

The syndrome of the received vector is calculated. It consists of syndromes of constituent-codes. If a constituent code contains erased symbols then its syndrome is not calculated and considered to be erased.

Error-suspicious symbols erasing



decision - value that should be added to the observed symbol to zero the constituent code syndrome.

c - number of null decisions.

e - number of erased decisions.

a - the subset of maximal cardinality containing equal neither zero nor erased decisions.

If $a > c + e$ than the symbol is replaced with erasure.

Correcting of erasures

For each symbol from the list of erased symbols the subset of constituent codes containing the symbol is considered. Only the codes containing one erasure belong to the subset. For each code from this subset we can correct the erasure and form a list of possible symbol values. Then the most often value is found and the erased symbol is replaced with this value.

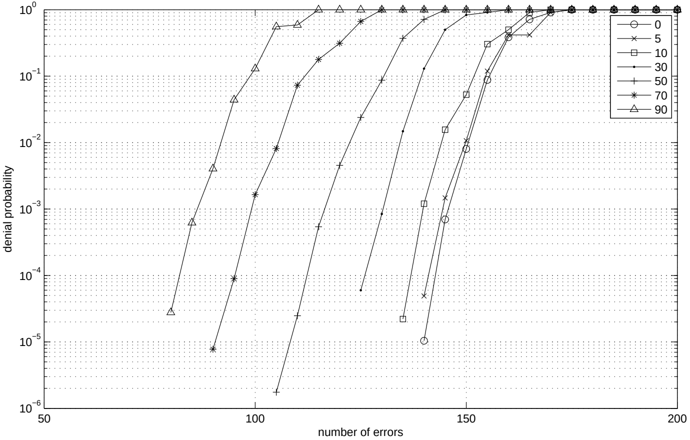
Stop criteria

Added erasures are removed. The syndromes before and after iteration are compared. Return to Step 2 in case of not equal syndromes. In case of equal syndromes the syndrome weight is calculated. If the weight is equal to zero then the decoded vector is returned. Return denial of decoding if the weight is not equal to zero.

Modeling description and chosen code parameters

We used a code with such parameters while modeling process: $q = 16$; $n = 2048$; $R = \frac{1}{2}$; $n_0 = 16$; $\ell = 8$. Modeling process starts from 300 errors. This value decreases by 5 errors after 10 denials are got. The result of a modeling process is a dependency of denial probability on number of errors (number of erasures is fixed). Over than 10^6 tests are carried out for each dependency.

The dependency of realized correcting capabilities on initial number of erasures

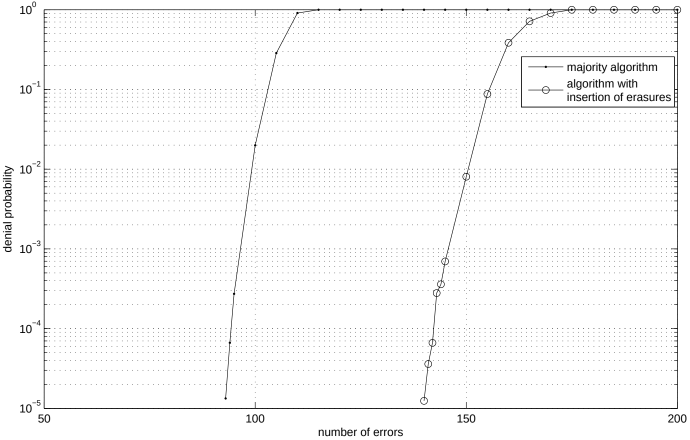


The dependency of realized correcting capabilities on initial number of erasures

Let τ denote the number of erasures, e^* denote the number of errors in case of the denial probability is less than 10^{-4} (the greatest number of errors which meets the condition is chosen). We will use a value $d^* = 2e^* + \tau + 1$ to describe the realized correcting capabilities of the suggested algorithm. The dependency of realized correcting capabilities of the suggested algorithm on initial number of erasures is introduced in the Table.

τ	0	5	10	30	50	70	90
e^*	142	140	136	126	110	94	81
d^*	285	286	283	283	271	259	253

The comparison with a majority algorithm



Results

1. An iterative decoding algorithm capable of correcting both errors and erasures is developed.
2. The dependency of realized correcting capabilities of the algorithm on initial number of erasures is introduced.
3. Error-correcting capabilities of the algorithm are better than error-correcting capabilities of a majority algorithm.

Futher research

It is possible to use this algorithm in case of more powerful constituent code without any changes. But more powerful constituent codes can correct to $d-1$ erasures and it is rational to change the decoding algorithm.

Thank you for the attention!