
Extended binary image of non-binary LDPC codes, and decoding applications

Valentin Savin

14 janvier 2010

GdR-ISIS, Paris

Non-Binary LDPC codes

Non-binary LDPC codes

$$\mathbf{F}_2 = \{0, 1\}, \quad q = 2^p$$

$$\text{Non-binary alphabet: } \mathbf{A} \cong \mathbf{F}_2^p : X \cong (x_0, \dots, x_{p-1})$$

\uparrow symbol \uparrow binary image

$$\mathbf{L} = \text{End}(\mathbf{A}) \cong \mathbf{M}_p(\mathbf{F}_2), \text{ acts on the left on } \mathbf{A}$$

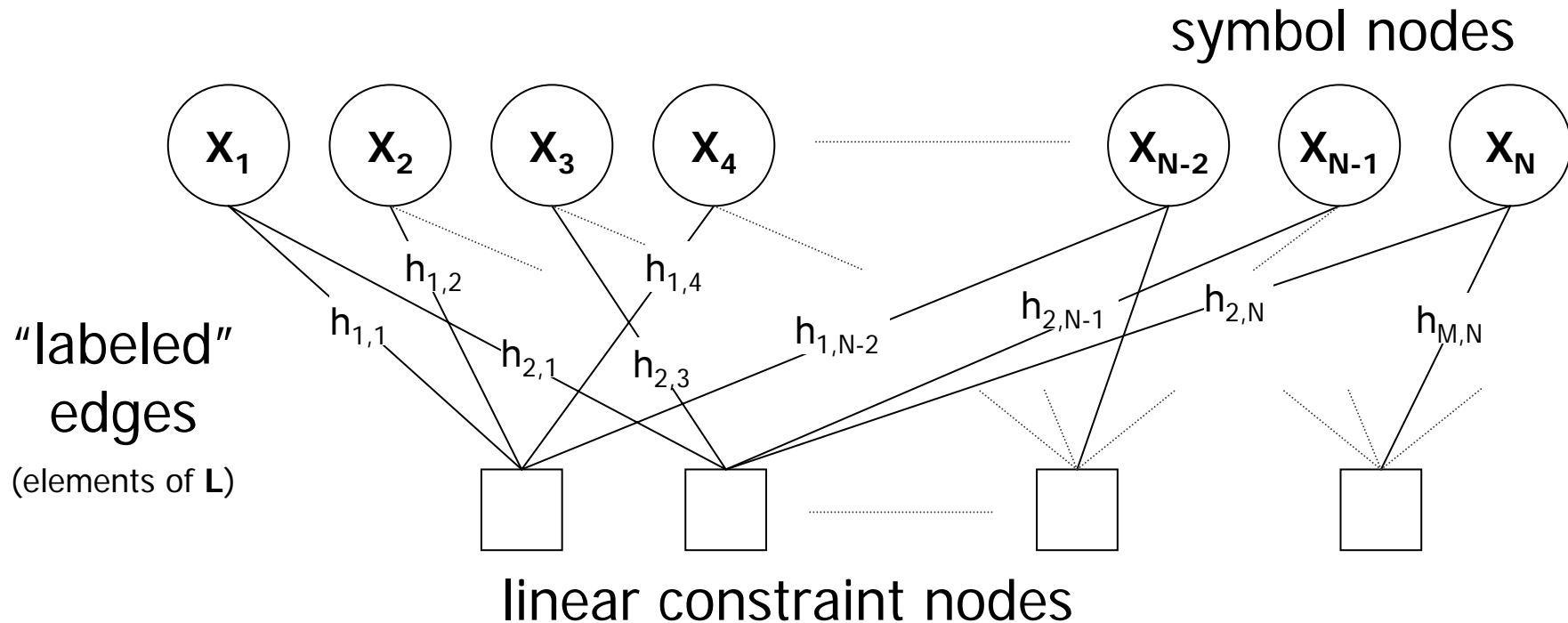
$$\mathbf{L} \times \mathbf{A} \rightarrow \mathbf{A} : (h, X) \rightarrow h.X$$

$$H \in \mathbf{M}_{M,N}(\mathbf{L})$$

$$\mathcal{C} = \text{Ker}(H)$$

$$= \{ (X_1, X_2, \dots, X_N) \mid \sum_n (h_{m,n} X_n) = 0, \forall m = 1, \dots, M \}$$

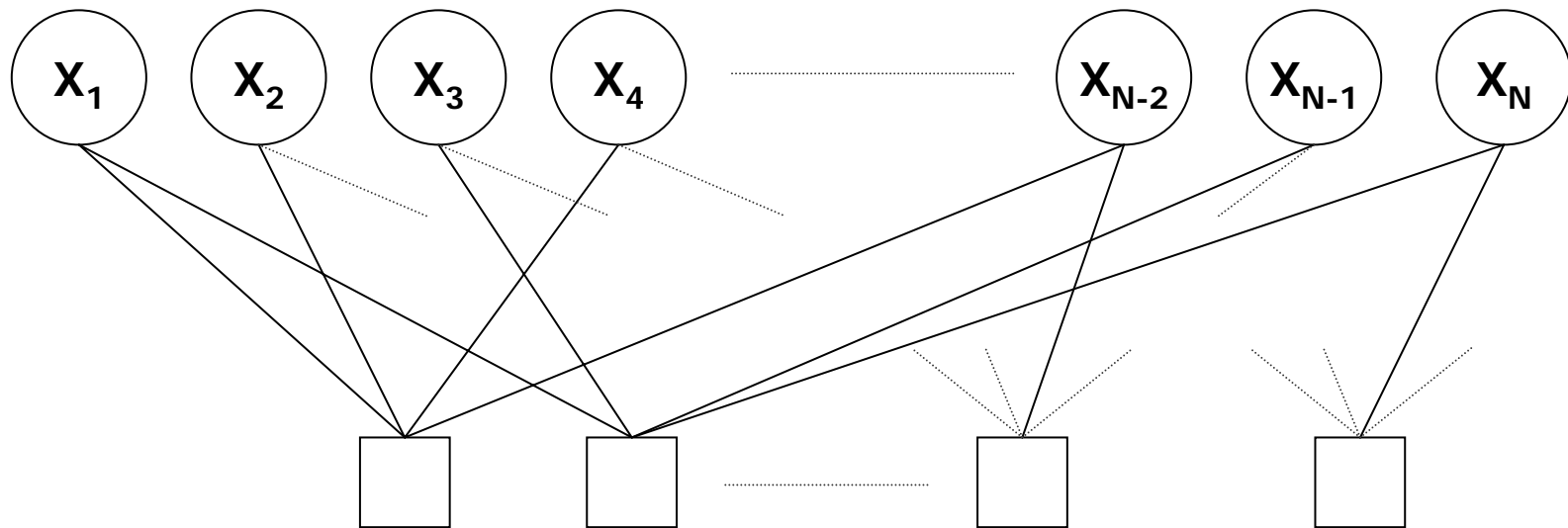
Non-binary LDPC codes



Codewords: solutions of a linear system with coefficients in \mathbf{L}

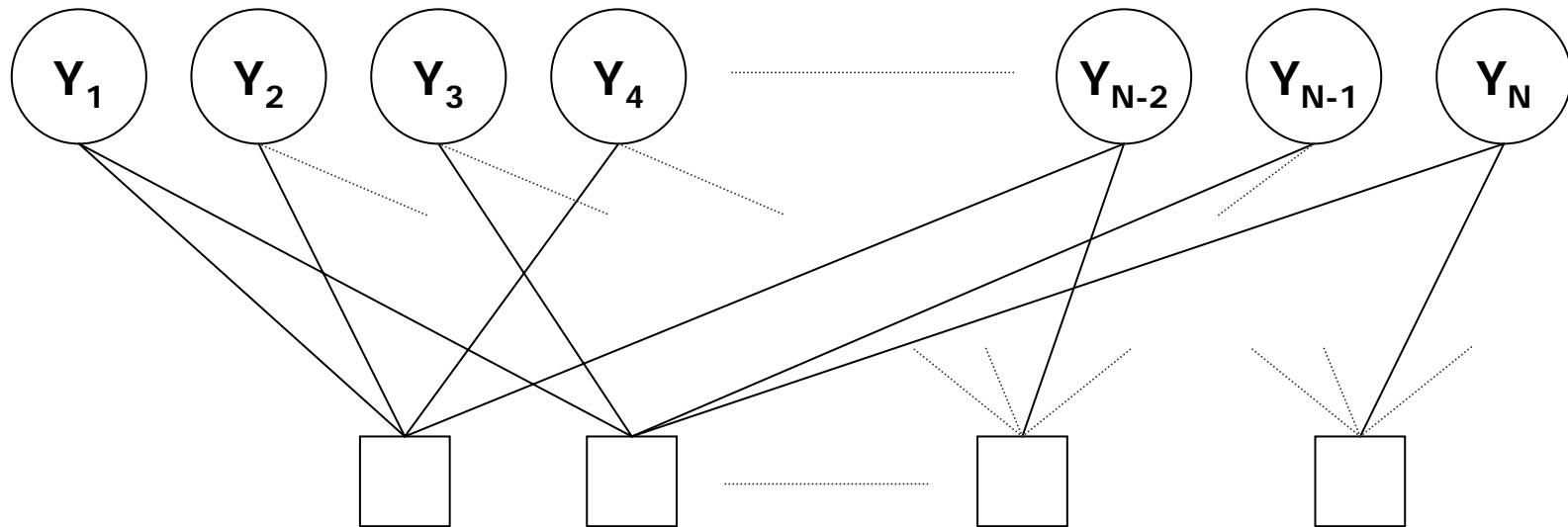
$$\begin{cases} h_{1,1}X_1 + h_{1,2}X_2 + h_{1,4}X_4 + h_{1,N-2}X_{N-2} = 0 \\ h_{2,1}X_1 + h_{2,3}X_3 + h_{2,N-1}X_{N-1} + h_{2,N}X_N = 0 \\ \dots \end{cases}$$

Non-binary LDPC codes



Codeword: X_1, X_2, \dots, X_N

Non-binary LDPC codes



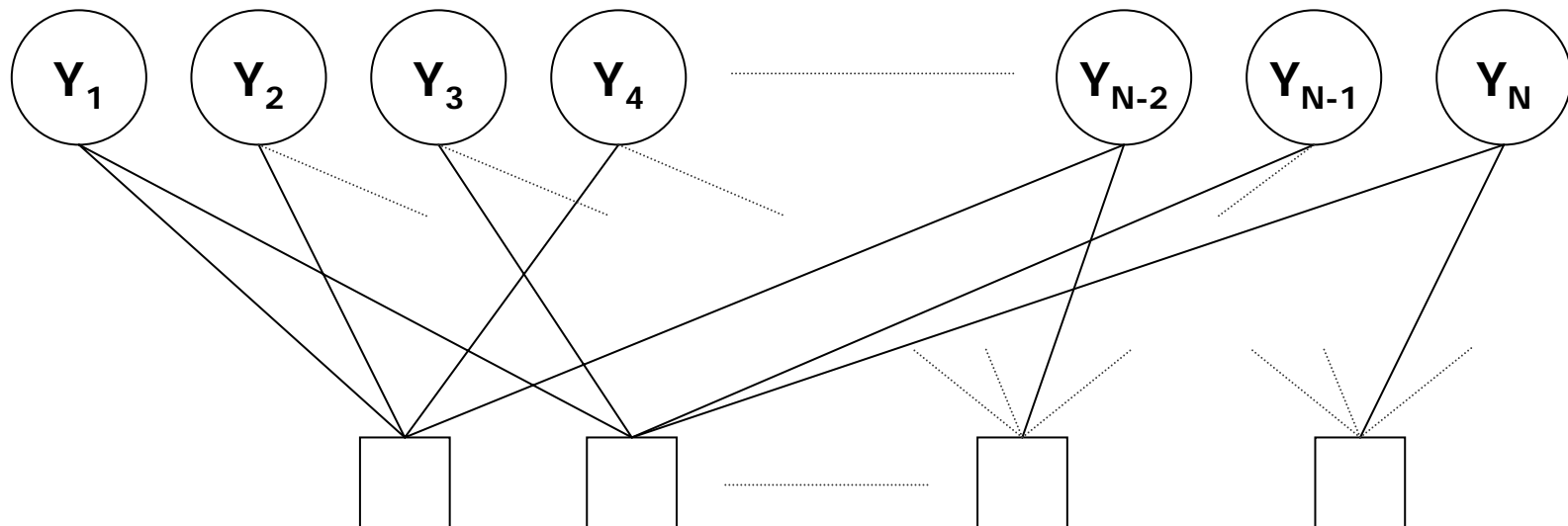
Codeword: X_1, X_2, \dots, X_N

Received : Y_1, Y_2, \dots, Y_N

binary channels: $X_n = (x_{n,0}, \dots, x_{n,p-1}) \rightarrow Y_n = (y_{n,0}, \dots, y_{n,p-1})$

Belief-Propagation Decoding

$$\gamma_1 = [\gamma_1(X)]_{X \in \mathcal{A}}, \dots, \gamma_n = [\gamma_n(X)]_{X \in \mathcal{A}}, \dots, \gamma_N = [\gamma_N(X)]_{X \in \mathcal{A}}$$

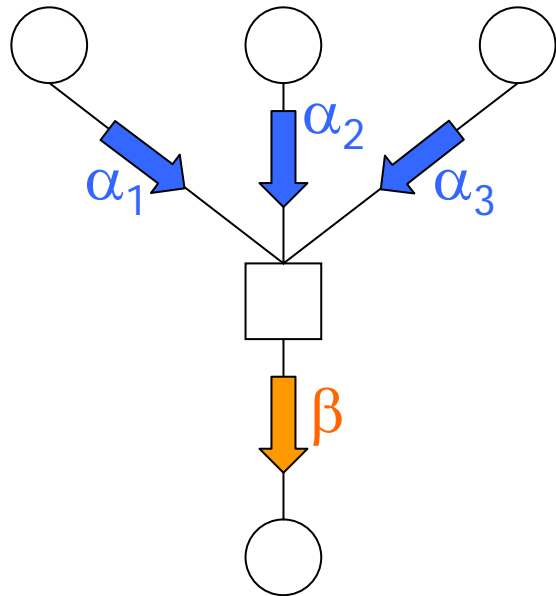


$$\gamma_n(X) = \Pr(X_n = X \mid Y_n), \quad X \in \mathcal{A}$$

(probability that \mathbf{X} has been sent at position \mathbf{n} conditioned on the channel output)

Belief-Propagation Decoding

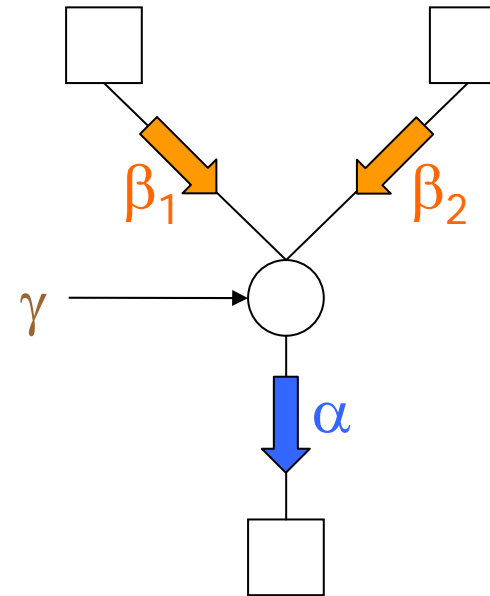
constraint-to-symbol node message



$$\beta(X) = \sum_{\substack{X_1, X_2, X_3 \\ h_1 X_1 + h_2 X_2 + h_3 X_3 + hX = 0}} \left(\prod \alpha_i(X_i) \right)$$

(first iteration: $\alpha_n = \gamma_n$)

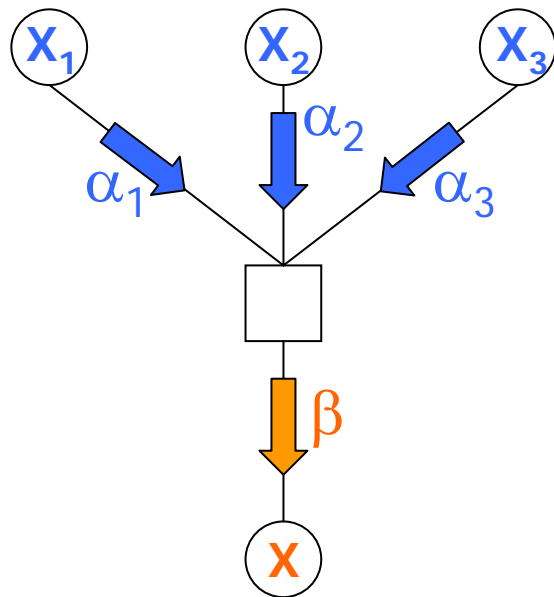
symbol-to-constraint node message



$$\alpha(X) = \gamma(X) \prod \beta_i(X)$$

Belief-Propagation Decoding

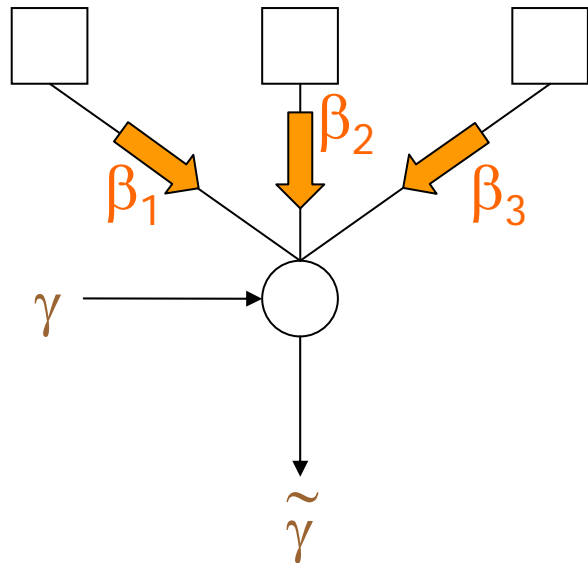
constraint-to-symbol node message



all X_1, X_2, X_3 , such that
 $h_1 X_1 + h_2 X_2 + h_3 X_3 + hX = 0$

$$\beta(X) = \sum_{\substack{X_1, X_2, X_3 \\ h_1 X_1 + h_2 X_2 + h_3 X_3 + hX = 0}} \left[\prod \alpha_i (X_i) \right]$$

Belief-Propagation Decoding



Stop if:

- all constraint-nodes are verified (successful decoding)
- or, maximum number of decoding iterations is reached

$$\tilde{\gamma}(X) = \gamma(X) \prod \beta_i(X) \Rightarrow \tilde{X} = \operatorname{argmax}(\tilde{\gamma}(X))$$

("updated" PDF)

Extended binary representation

Binary image of a non-binary LDPC code

Example:

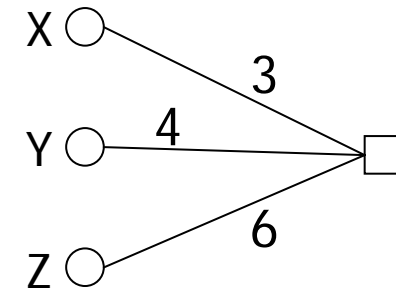
- non-binary constraint over \mathbf{F}_8

$$3X + 4Y + 6Z = 0$$

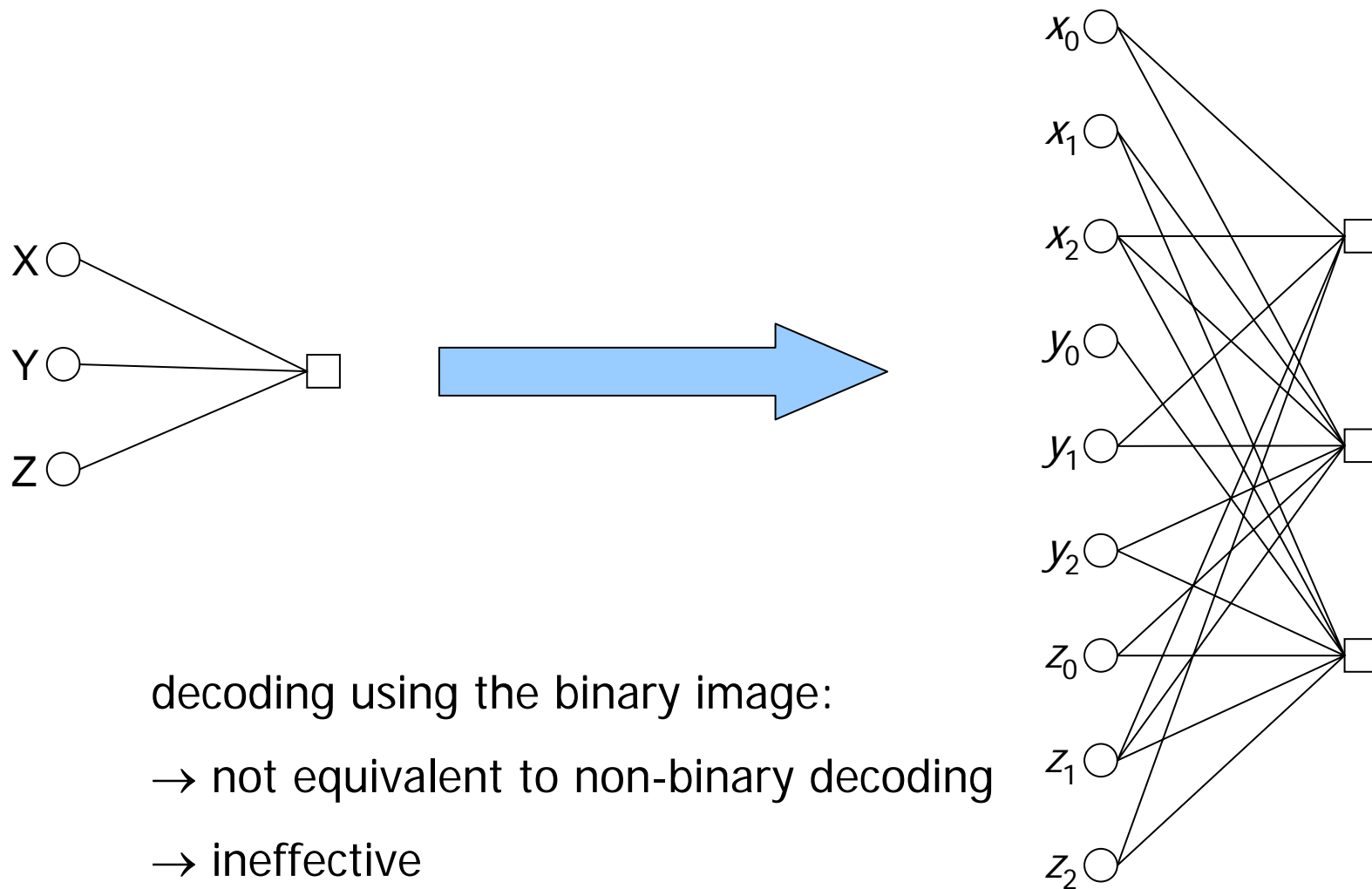
- binary image:

$$\begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} y_0 \\ y_1 \\ y_2 \end{pmatrix} + \begin{pmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} z_0 \\ z_1 \\ z_2 \end{pmatrix} = 0$$

$$\begin{cases} x_0 + x_2 + y_1 + z_1 + z_2 = 0 \\ x_0 + x_1 + x_2 + y_1 + y_2 + z_0 + z_1 = 0 \\ x_1 + x_2 + y_0 + y_2 + z_0 + z_1 + z_2 = 0 \end{cases}$$



Binary image of a non-binary LDPC code



Extended binary image

- Introduce extra binary nodes, representing all the linear combinations of bits x_i

$$(a_1, a_2, a_3, a_4, a_5, a_6, a_7) = (x_0, x_1, x_2) \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

$$a_1 = x_0$$

$$a_2 = x_1$$

$$a_3 = x_0 + x_1$$

$$a_4 = x_2$$

$$a_5 = x_0 + x_2$$

$$a_6 = x_1 + x_2$$

$$a_7 = x_0 + x_1 + x_2$$

Extended binary image

- Introduce extra binary nodes, representing all the linear combinations of bits x_i

$$(a_1, a_2, a_3, a_4, a_5, a_6, a_7) = (x_0, x_1, x_2) \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

extended
binary image

(simplex codeword)

$$a_1 = x_0$$

$$a_2 = x_1$$

$$a_3 = x_0 + x_1$$

$$a_4 = x_2$$

$$a_5 = x_0 + x_2$$

$$a_6 = x_1 + x_2$$

$$a_7 = x_0 + x_1 + x_2$$

binary image

Extended binary image

$$\begin{cases} x_0 + x_2 + y_1 + z_1 + z_2 = 0 \\ x_0 + x_1 + x_2 + y_1 + y_2 + z_0 + z_1 = 0 \\ x_1 + x_2 + y_0 + y_2 + z_0 + z_1 + z_2 = 0 \end{cases}$$

Extended binary image

$$\begin{cases} (x_0 + x_2) + y_1 + (z_1 + z_2) = 0 \\ (x_0 + x_1 + x_2) + (y_1 + y_2) + (z_0 + z_1) = 0 \\ (x_1 + x_2) + (y_0 + y_2) + (z_0 + z_1 + z_2) = 0 \end{cases}$$

$$\begin{cases} a_5 + b_2 + c_6 = 0 \\ a_7 + b_6 + c_3 = 0 \\ a_6 + b_5 + c_7 = 0 \end{cases}$$

Extended binary image

extended
binary image

a_1	a_2	a_3	a_4	a_5	a_6	a_7	b_1	b_2	b_3	b_4	b_5	b_6	b_7	c_1	c_2	c_3	c_4	c_5	c_6	c_7
			1				1												1	
						1						1			1					
					1					1										1

Extended binary image

extended
binary image

a_1	a_2	a_3	a_4	a_5	a_6	a_7	b_1	b_2	b_3	b_4	b_5	b_6	b_7	c_1	c_2	c_3	c_4	c_5	c_6	c_7	
			1				1													1	
						1						1			1						
					1					1											1

- no cycles
- but they are unconstrained **extended binary variables** a_i, b_i, c_i

Extended binary image

- Introduce extra parity checks: all possible combinations of original parity-checks

$$\left\{ \begin{array}{l} a_5 + b_2 + c_6 = 0 \\ a_7 + b_6 + c_3 = 0 \\ a_6 + b_5 + c_7 = 0 \end{array} \right\} \Rightarrow (a_5 + a_7) + (b_2 + b_6) + (c_6 + c_3) = 0$$

Extended binary image

- Introduce extra parity checks: all possible combinations of original parity-checks

$$\left\{ \begin{array}{l} a_5 + b_2 + c_6 = 0 \\ a_7 + b_6 + c_3 = 0 \\ a_6 + b_5 + c_7 = 0 \end{array} \right\} \Rightarrow (a_5 + a_7) + (b_2 + b_6) + (c_6 + c_3) = 0$$
$$\Downarrow$$
$$a_2 + b_4 + c_5 = 0$$

$$a_1 = x_0$$

$$a_2 = x_1$$

$$a_3 = x_0 + x_1$$

$$a_4 = x_2$$

$$a_5 = x_0 + x_2$$

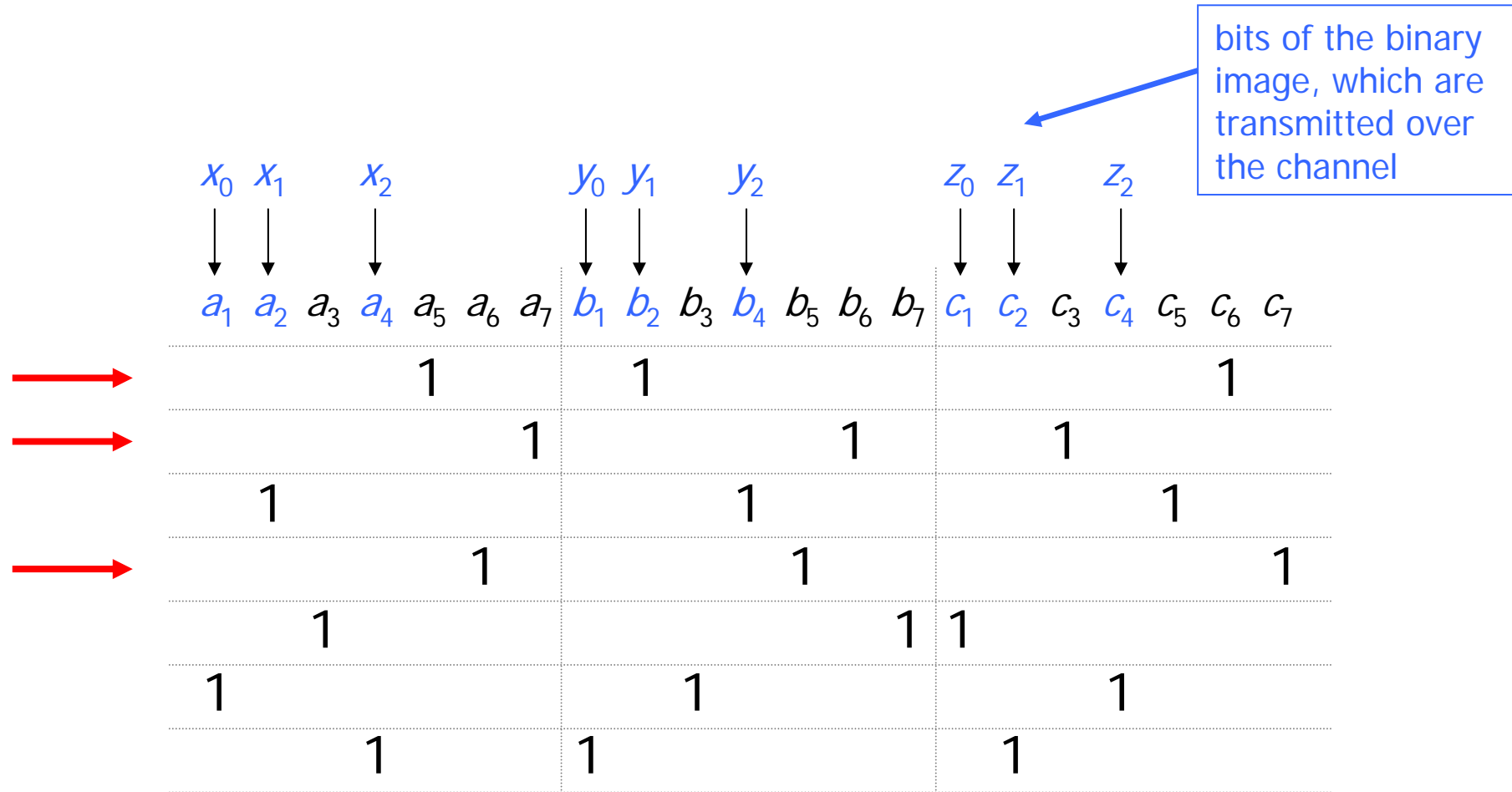
$$a_6 = x_1 + x_2$$

$$a_7 = x_0 + x_1 + x_2$$

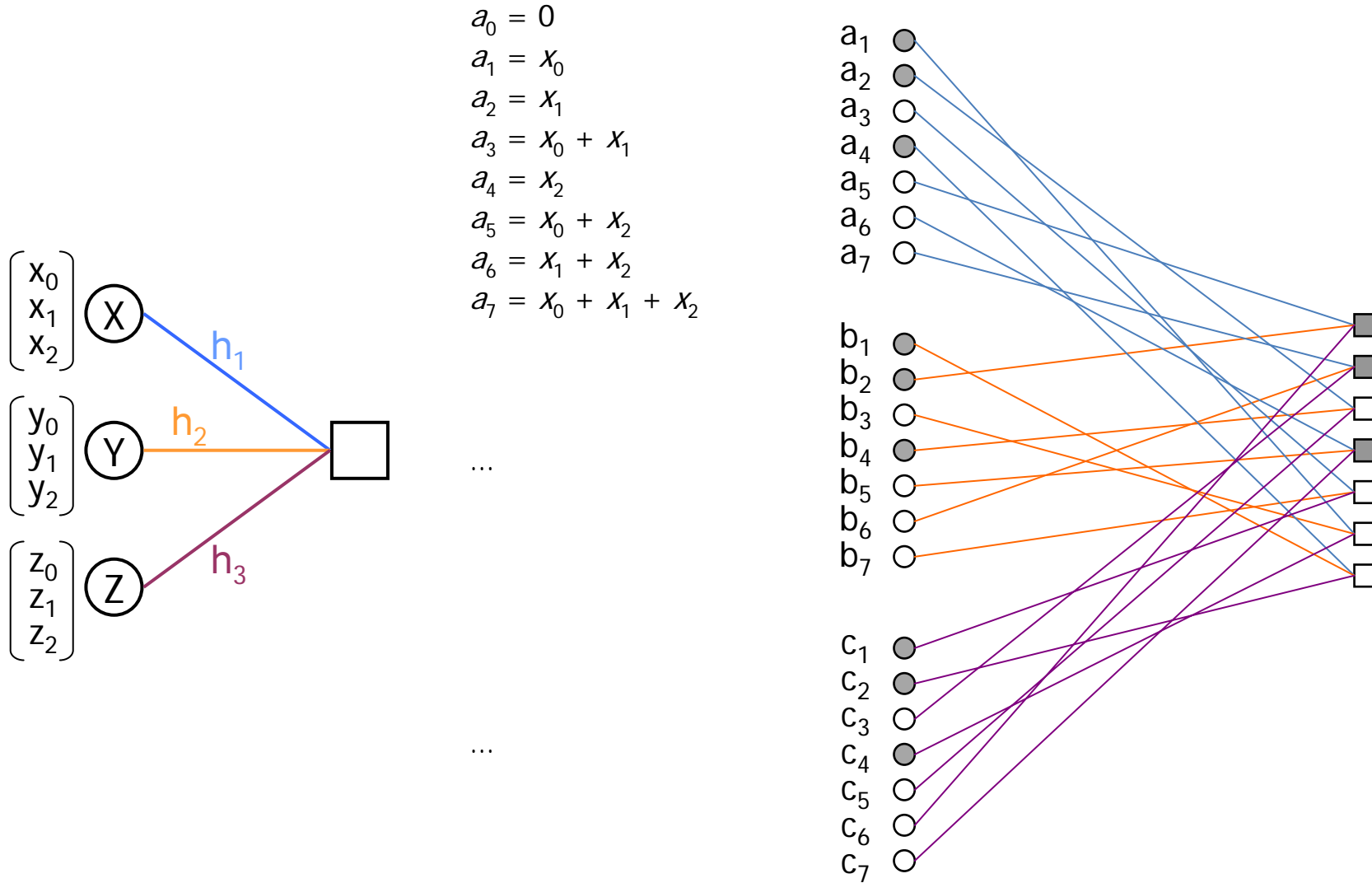
Extended binary image

a_1	a_2	a_3	a_4	a_5	a_6	a_7	b_1	b_2	b_3	b_4	b_5	b_6	b_7	c_1	c_2	c_3	c_4	c_5	c_6	c_7
			1				1													1
						1						1			1					
1									1									1		
					1					1										1
	1												1	1						
1								1									1			
		1					1								1					

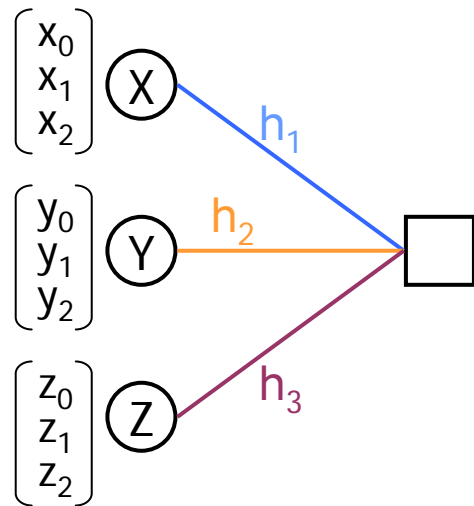
Extended binary image



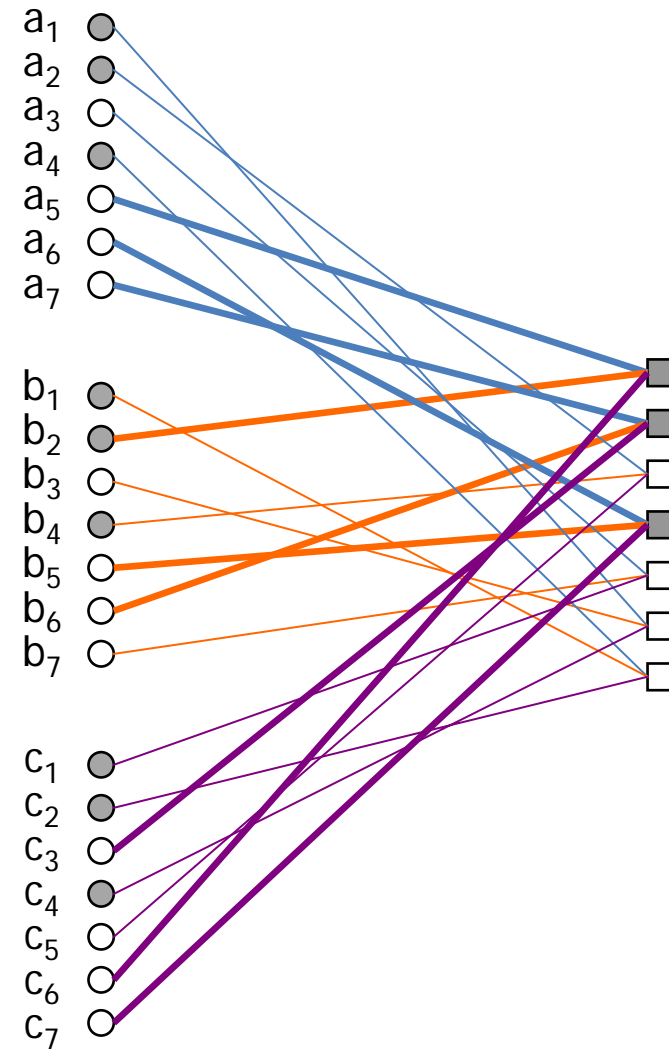
Extended binary graph



Extended binary graph

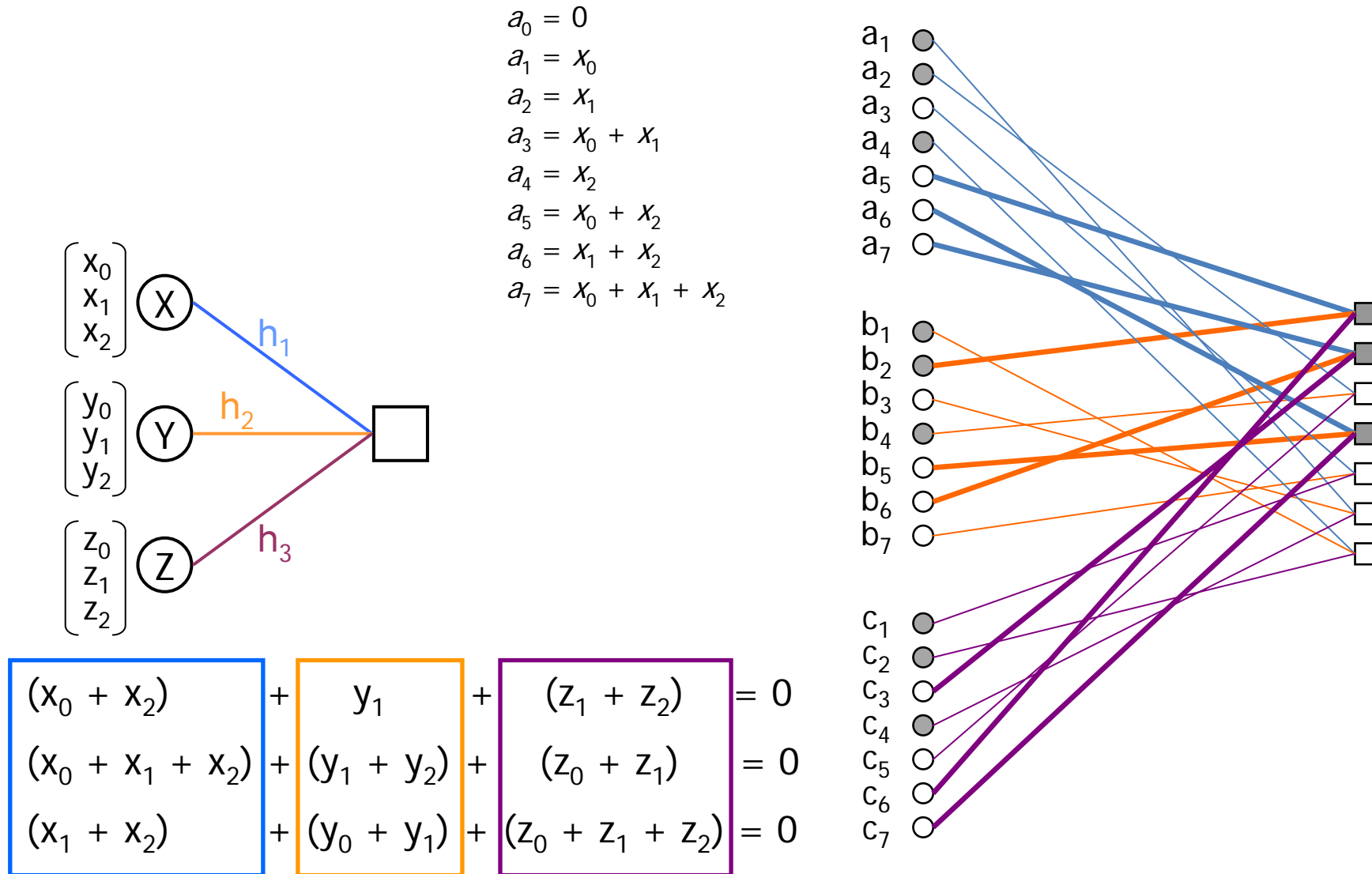


$$\begin{aligned}
 a_0 &= 0 \\
 a_1 &= x_0 \\
 a_2 &= x_1 \\
 a_3 &= x_0 + x_1 \\
 a_4 &= x_2 \\
 a_5 &= x_0 + x_2 \\
 a_6 &= x_1 + x_2 \\
 a_7 &= x_0 + x_1 + x_2
 \end{aligned}$$

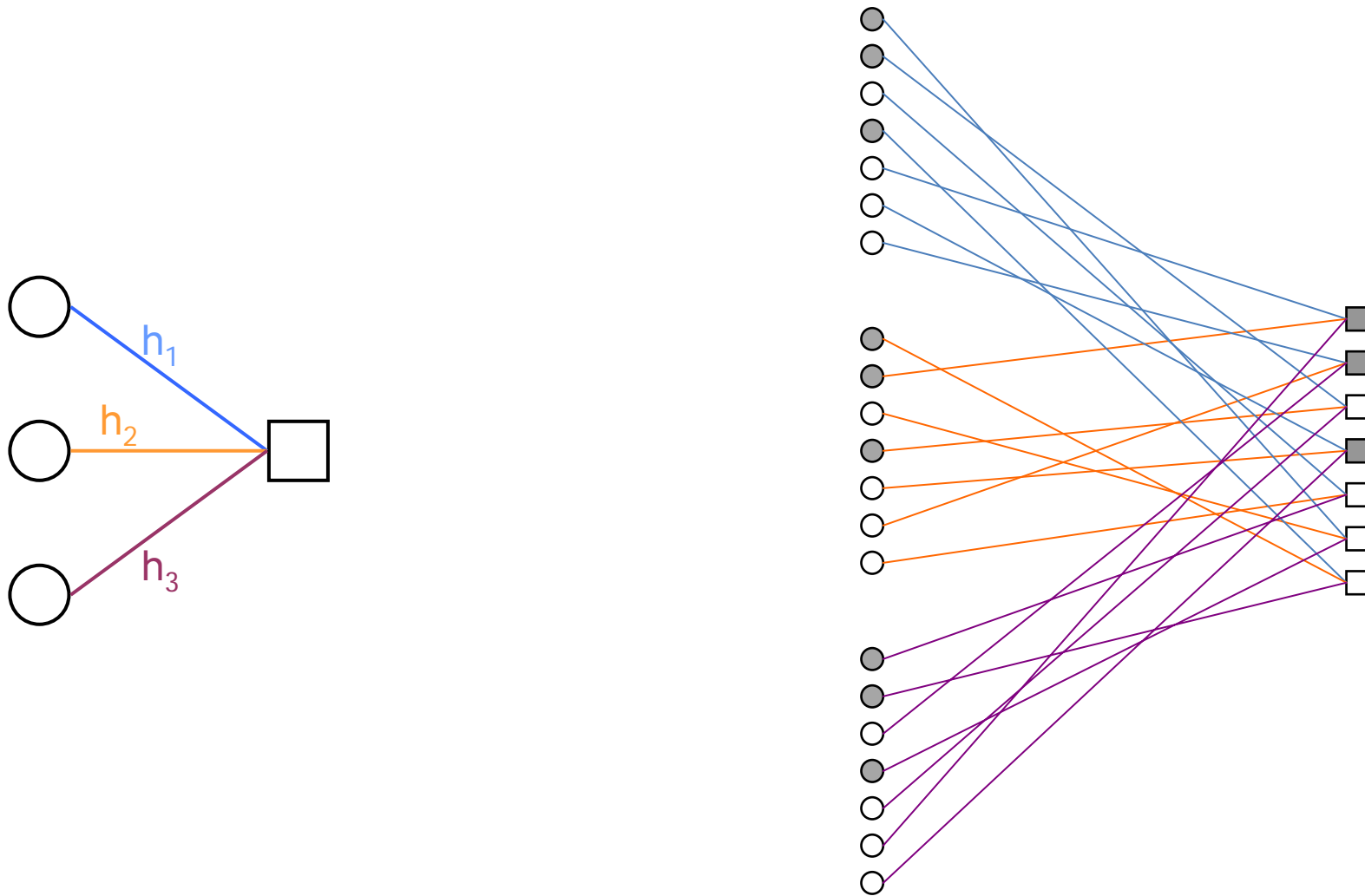


$$\begin{aligned}
 (x_0 + x_2) + y_1 + (z_1 + z_2) &= 0 \\
 (x_0 + x_1 + x_2) + (y_1 + y_2) + (z_0 + z_1) &= 0 \\
 (x_1 + x_2) + (y_0 + y_1) + (z_0 + z_1 + z_2) &= 0
 \end{aligned}$$

Extended binary graph

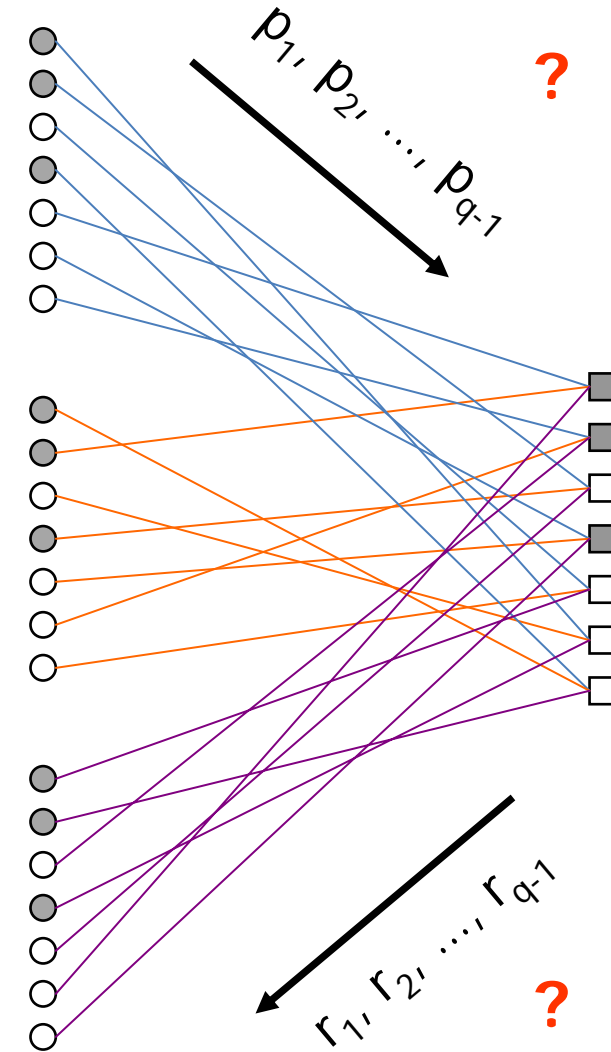
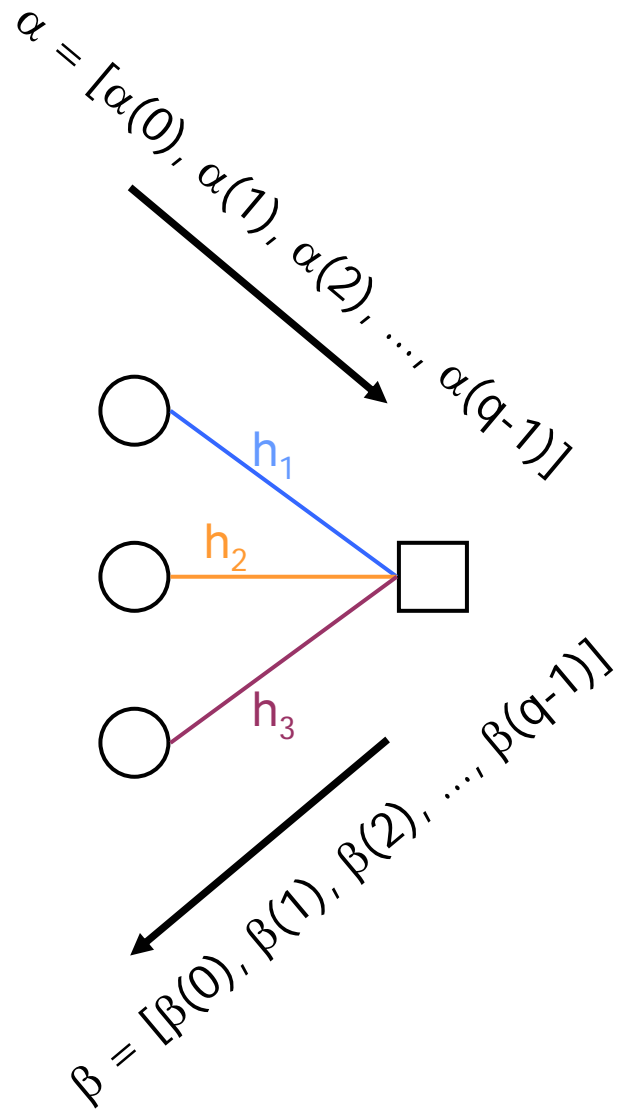


Extended binary graph



Decoding over general channels

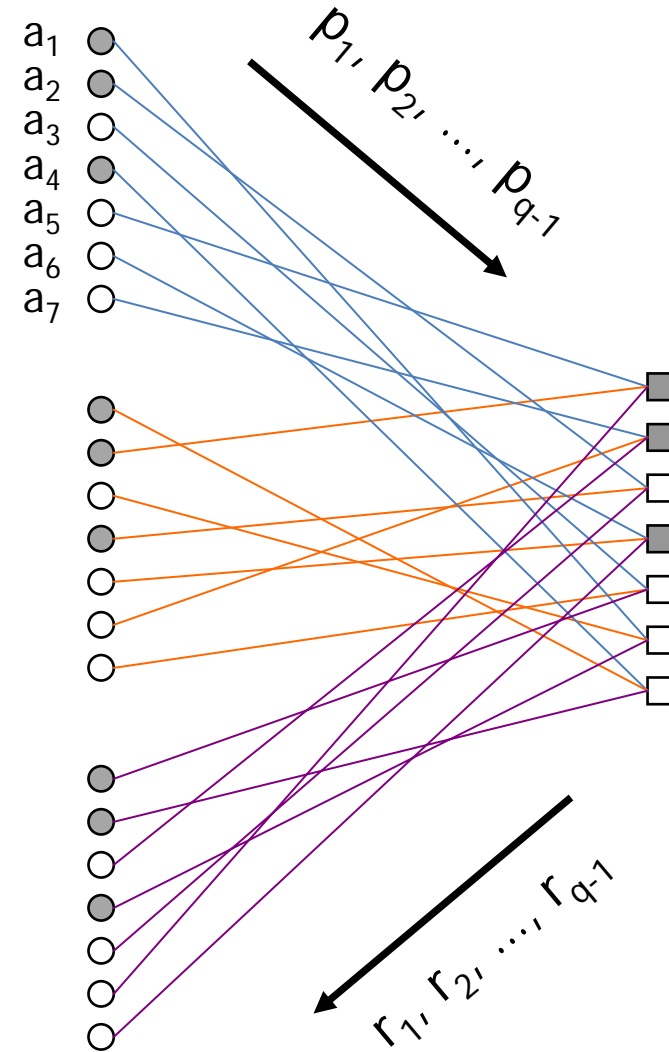
Exchanged messages



Exchanged messages

$$p_k = \Pr(a_k = 0) - \Pr(a_k = 1), k = 1, \dots, q-1$$

define also $p_0 = 1$



Exchanged messages

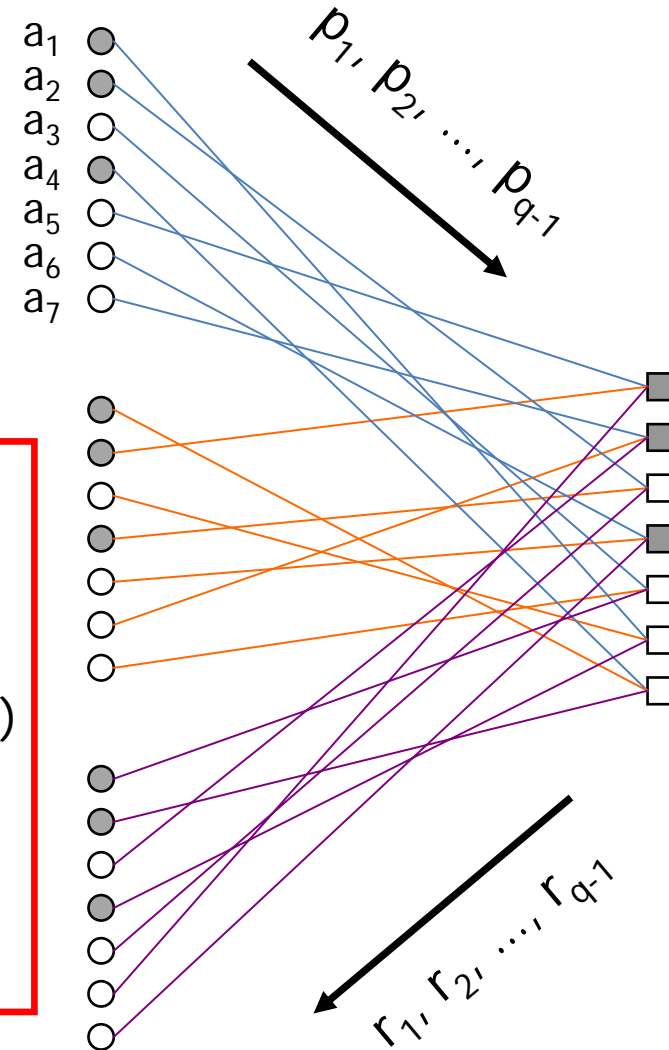
$$p_k = \Pr(a_k = 0) - \Pr(a_k = 1), \quad k = 1, \dots, q-1$$

define also $p_0 = 1$

Prop.

$$\begin{aligned} (p_0, p_1, \dots, p_{q-1}) &= \text{BFT}(\alpha) \\ &= \text{BFT}(\alpha(0), \alpha(1), \dots, \alpha(q-1)) \end{aligned}$$

$$\begin{aligned} (r_0, r_1, \dots, r_{q-1}) &= \text{BFT}(\beta) \\ &= \text{BFT}(\beta(0), \beta(1), \dots, \beta(q-1)) \end{aligned}$$



Iterative decoding using the extended binary image

■ Check-node processing

$$r_\ell = \prod p_{j,\ell_j}$$

$$\beta(S) = \sum (\prod \alpha_j(S_j))$$

where $\sum h_j S_j = hS$

■ Variable node processing

$$\beta = \text{IBFT}(r_0, r_1, \dots, r_{q-1})$$

$$\alpha(S) = \gamma(S) \cdot \prod \beta_i(S)$$

$$(p_0, p_1, \dots, p_{q-1}) = \text{BFT}(\alpha)$$

$$\alpha(S) = \gamma(S) \cdot \prod \beta_i(S)$$

Iterative decoding using the extended binary image

■ Check-node processing

$$r_\ell = \prod p_{j,\ell_j}$$

■ Variable node processing

$$p_k = \sum (\prod r_{i,k_i})$$

where $(\wedge k_i) = k$

bitwise XOR of integers k_i

$$\beta(S) = \sum (\prod \alpha_j(S_j))$$

where $\sum h_j S_j = hS$

$$\alpha(S) = \gamma(S) \cdot \prod \beta_i(S)$$

Extended binary graph

\cong

Fourier-domain graph

Binary Linear Time Erasure Decoding

(decoding over the binary erasure channel)

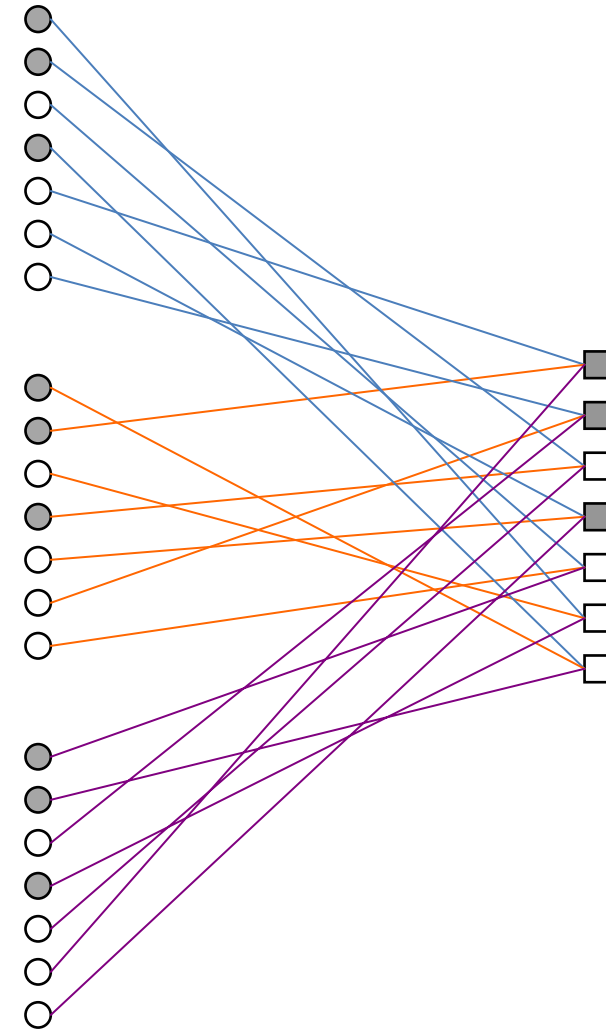
BLTE decoding

- check-node processing:

a parity-check that contains one single erased bit can recover its value

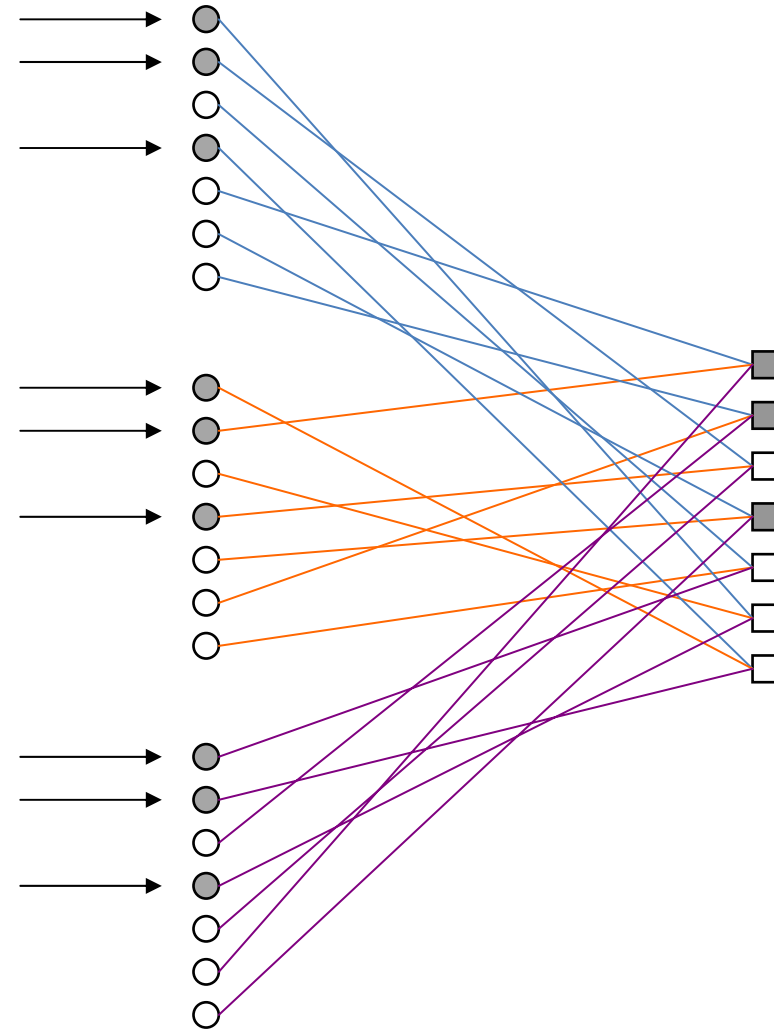
- simplex processing:

$$a_{i \wedge j \wedge \dots \wedge k} = a_i + a_j + \dots + a_k$$



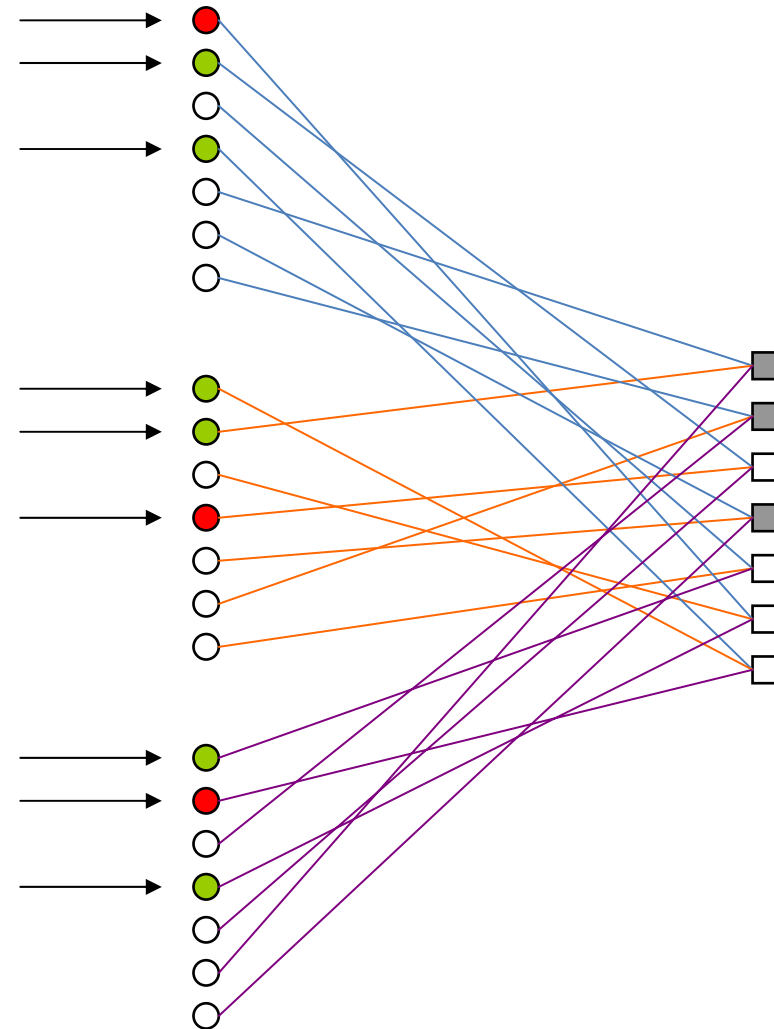
BLTE decoding

bits of the
binary
image
transmitted
over the
binary
erasure
channel

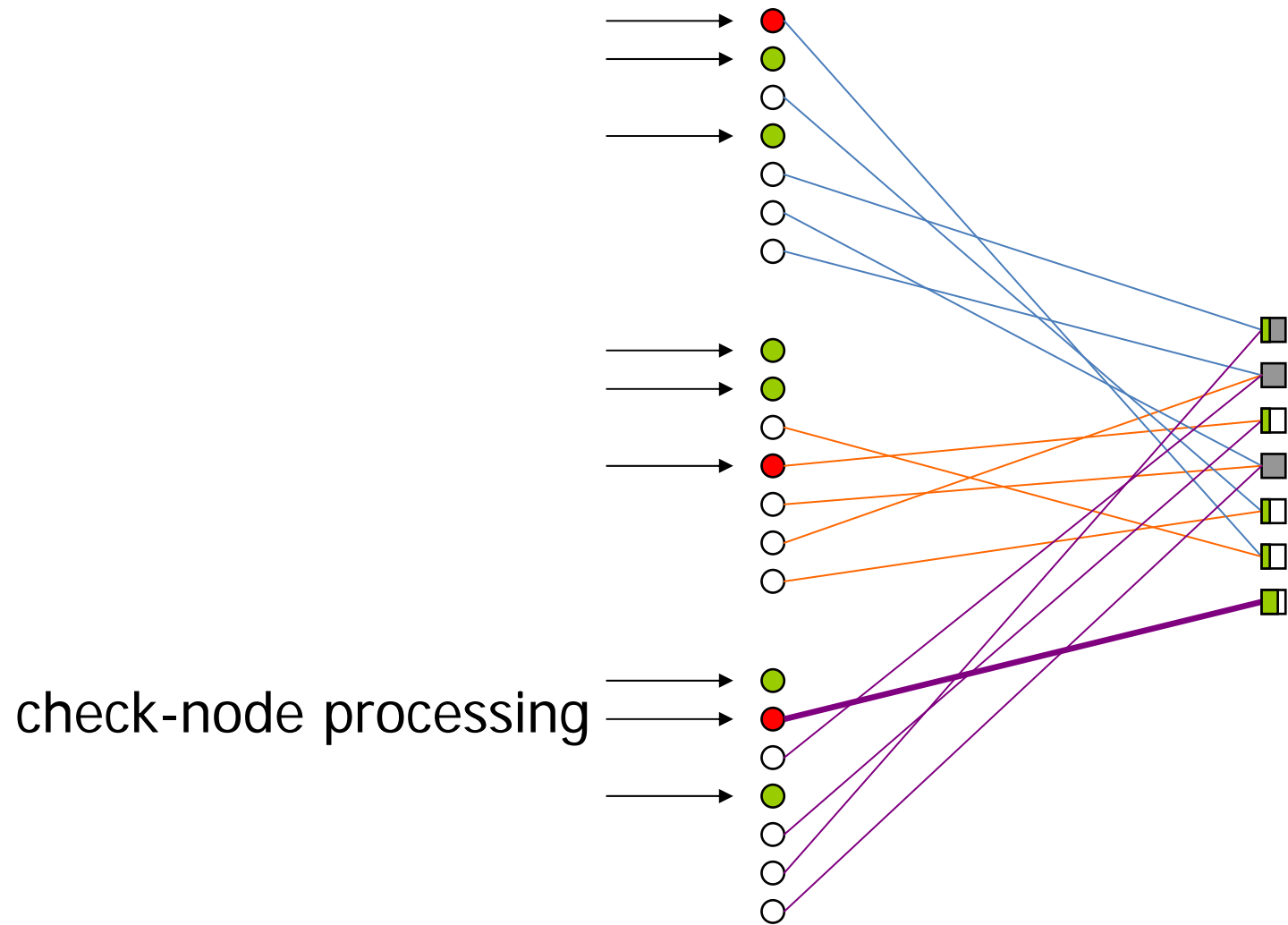


BLTE decoding

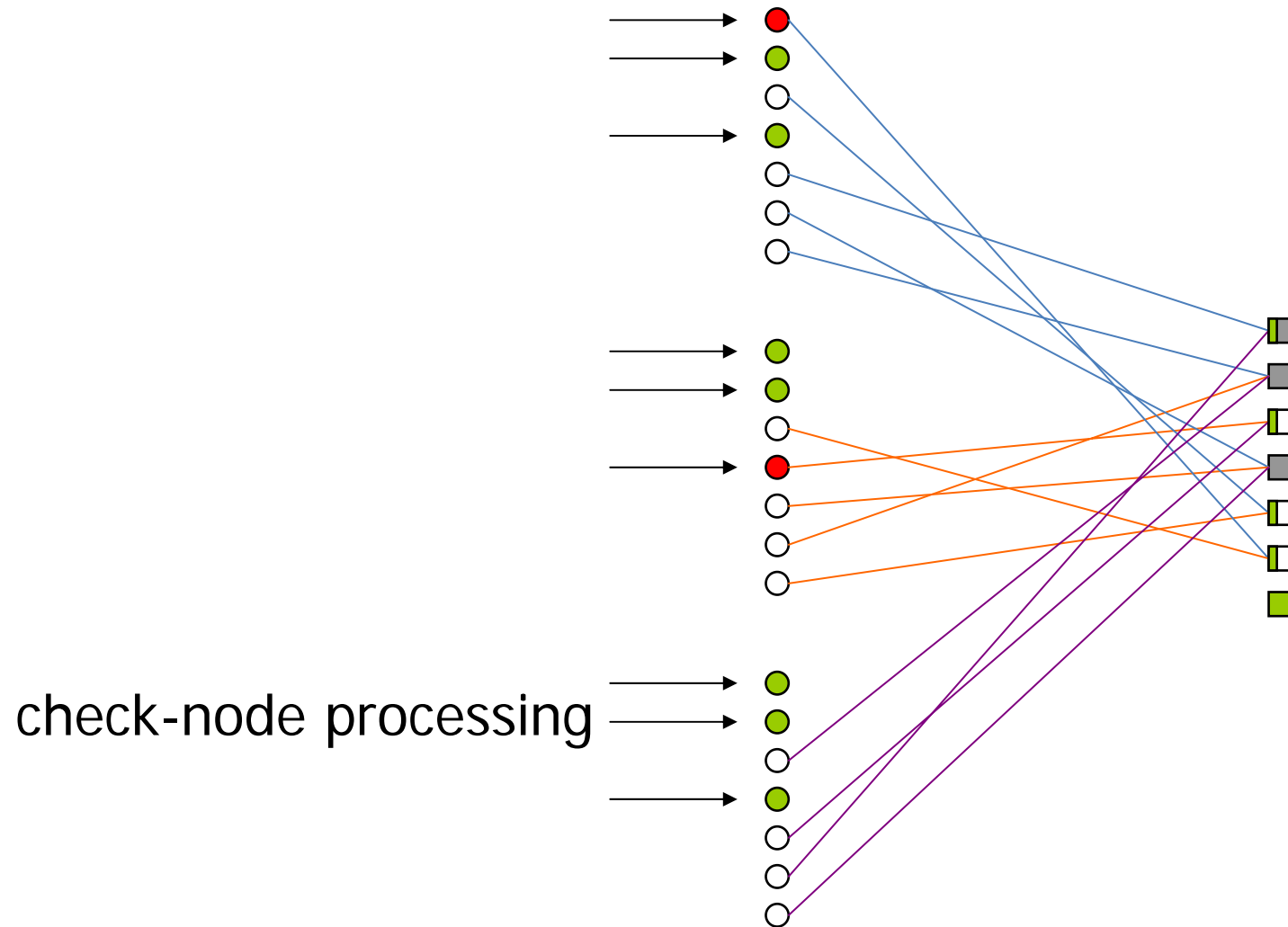
- erased
- received or recovered
- unknown (not transmitted)



BLTE decoding

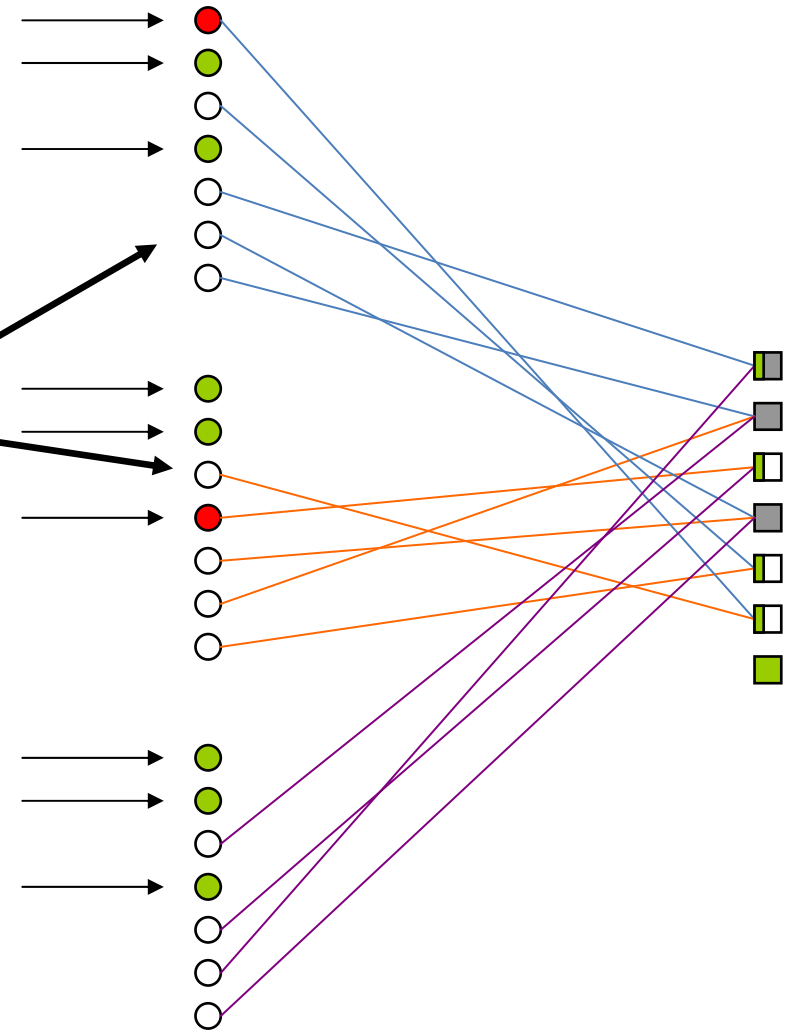


BLTE decoding



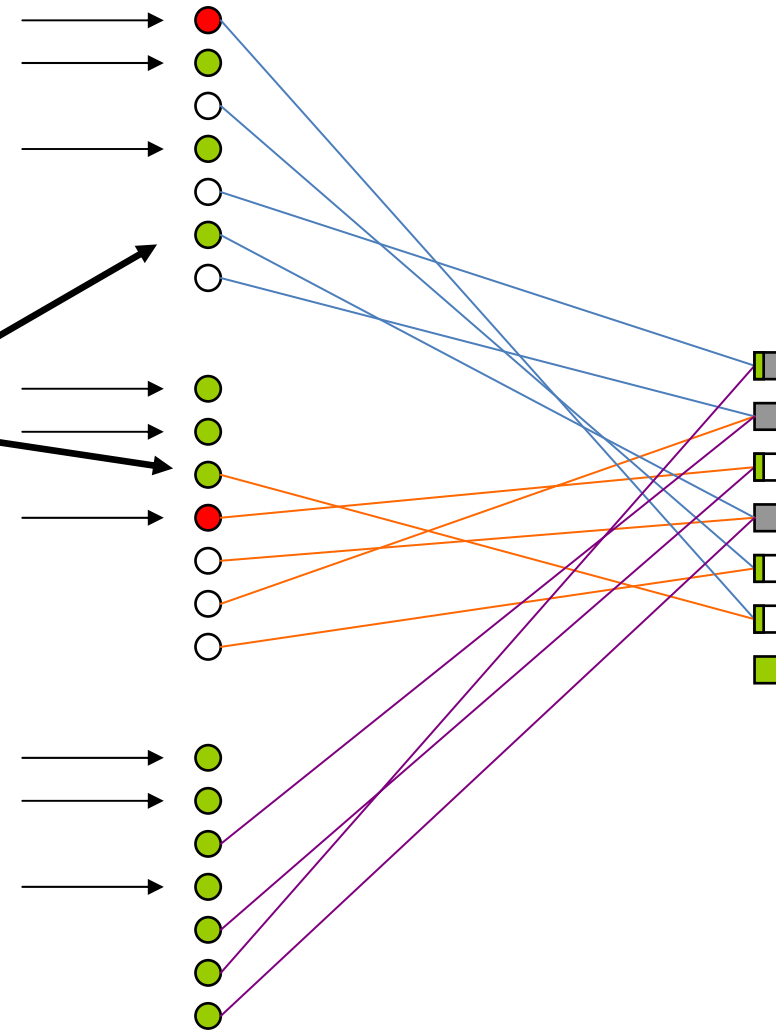
BLTE decoding

simplex processing:
recover all the bit-nodes
spanned by recovered nodes



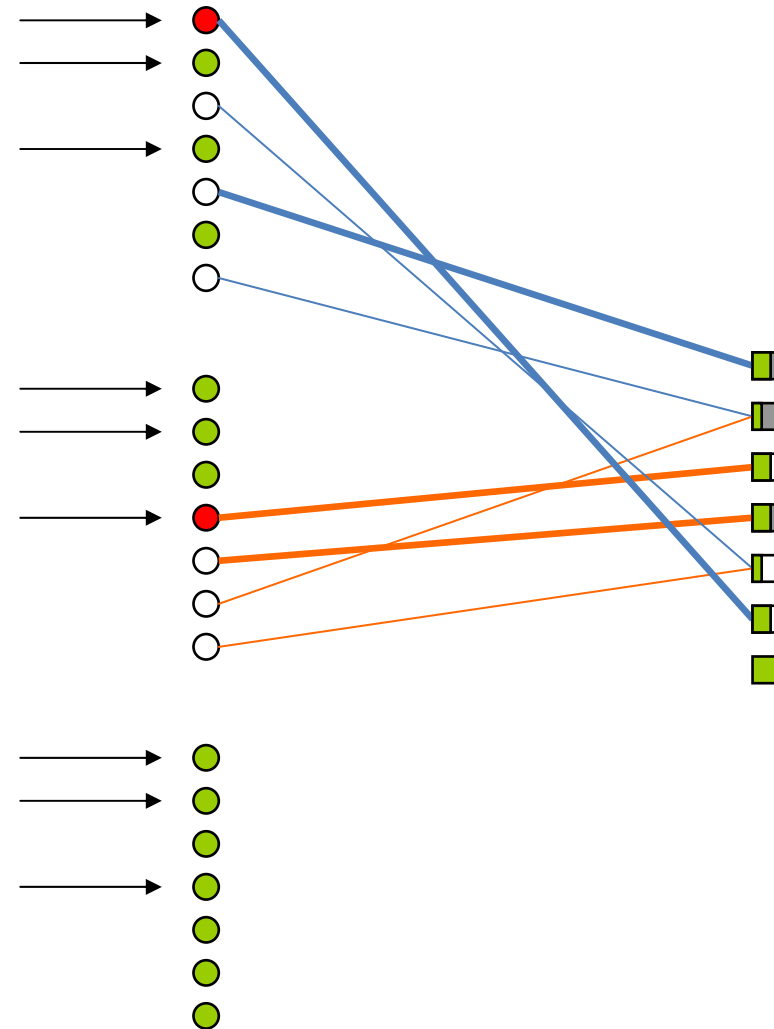
BLTE decoding

simplex processing:
recover all the bit-nodes
spanned by recovered nodes

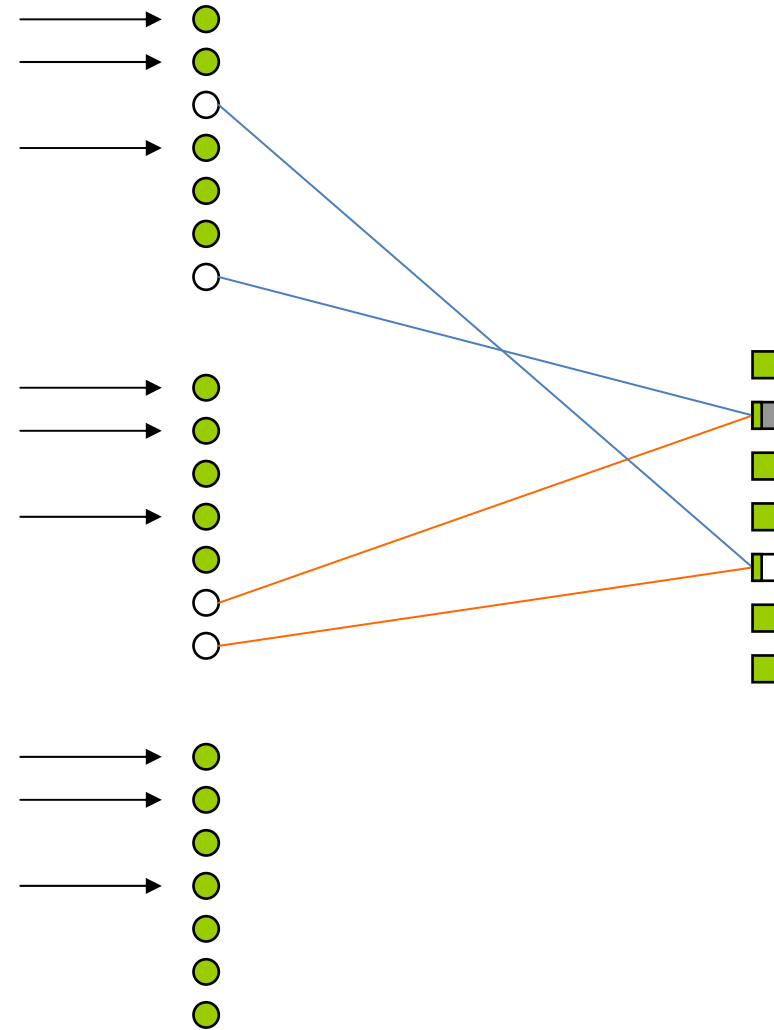


BLTE decoding

check-node processing

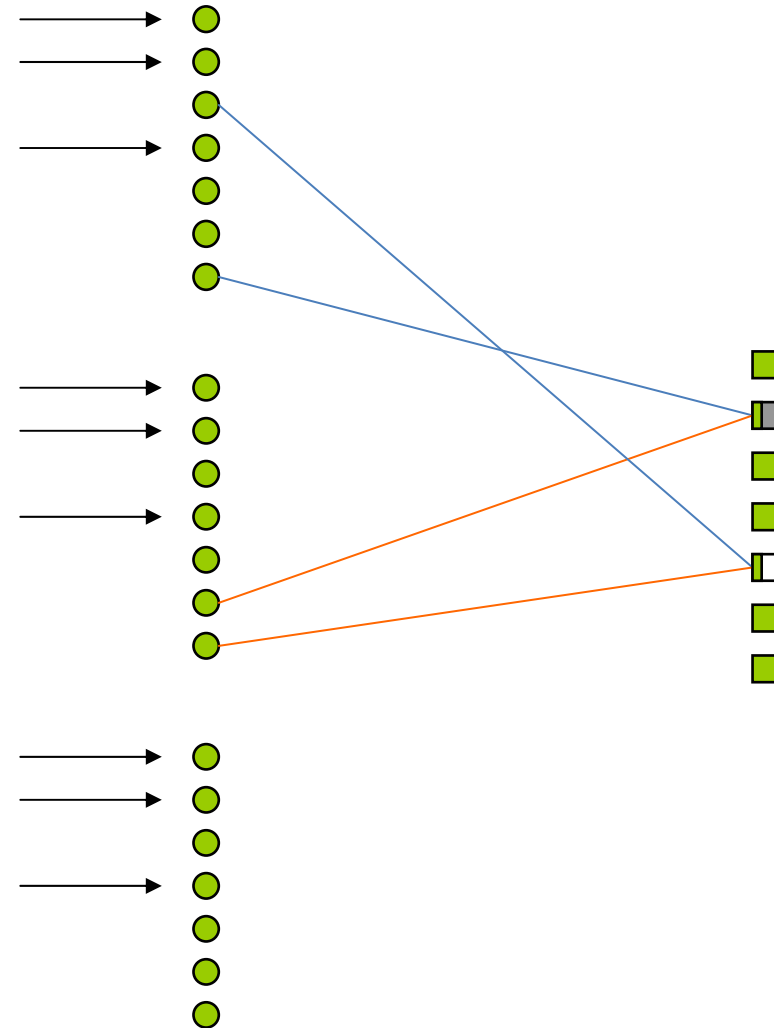


BLTE decoding



BLTE decoding

simplex processing:
recover all the bit-nodes
spanned by recovered nodes



BLTE decoding: complexity and performance

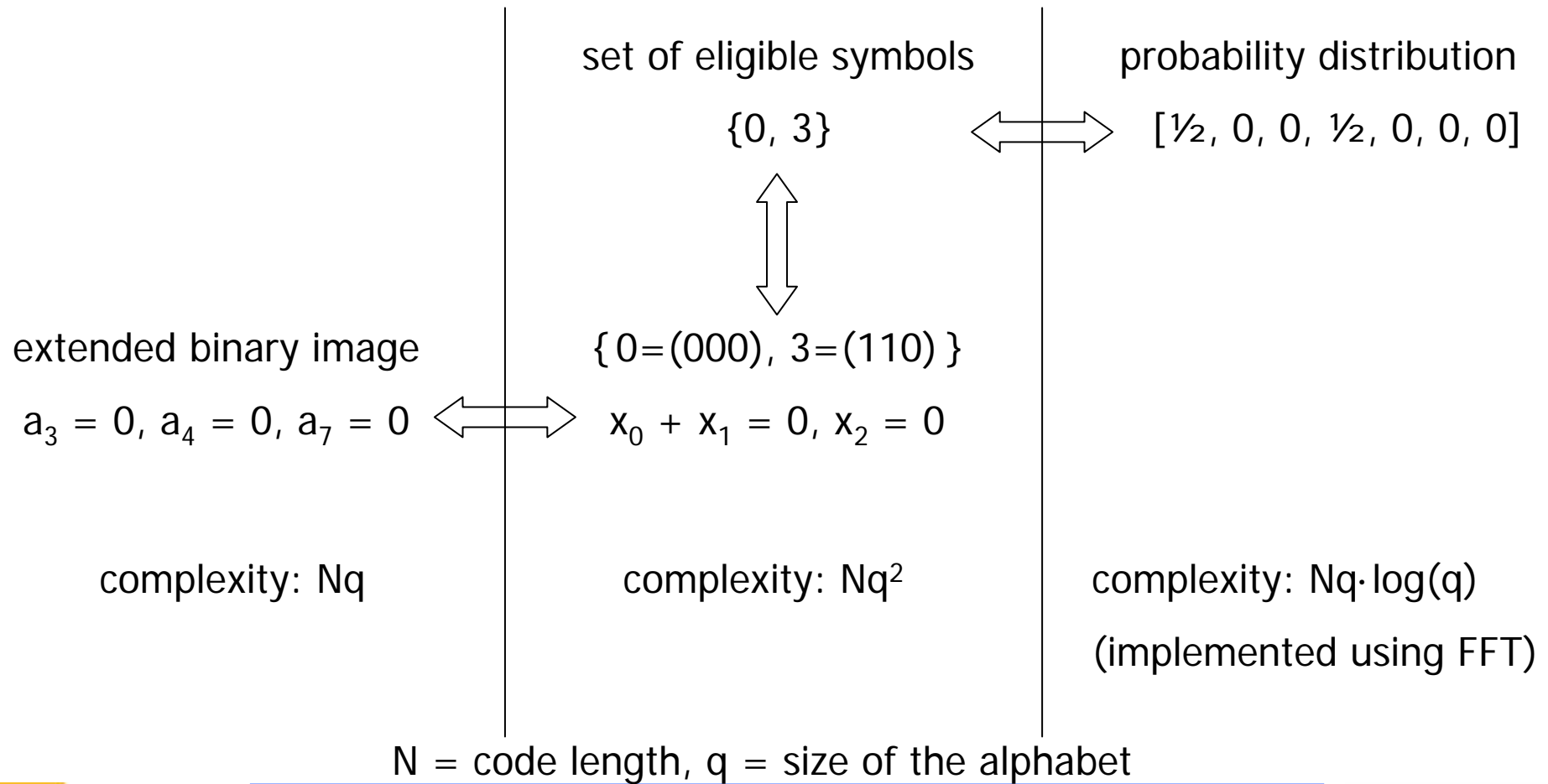
Theorem

- Complexity: $O(Nq)$
- Performance: BLTE \approx BP

Performance proof

BLTE Decoding \longleftrightarrow Erasure Decoding \longleftrightarrow BP decoding

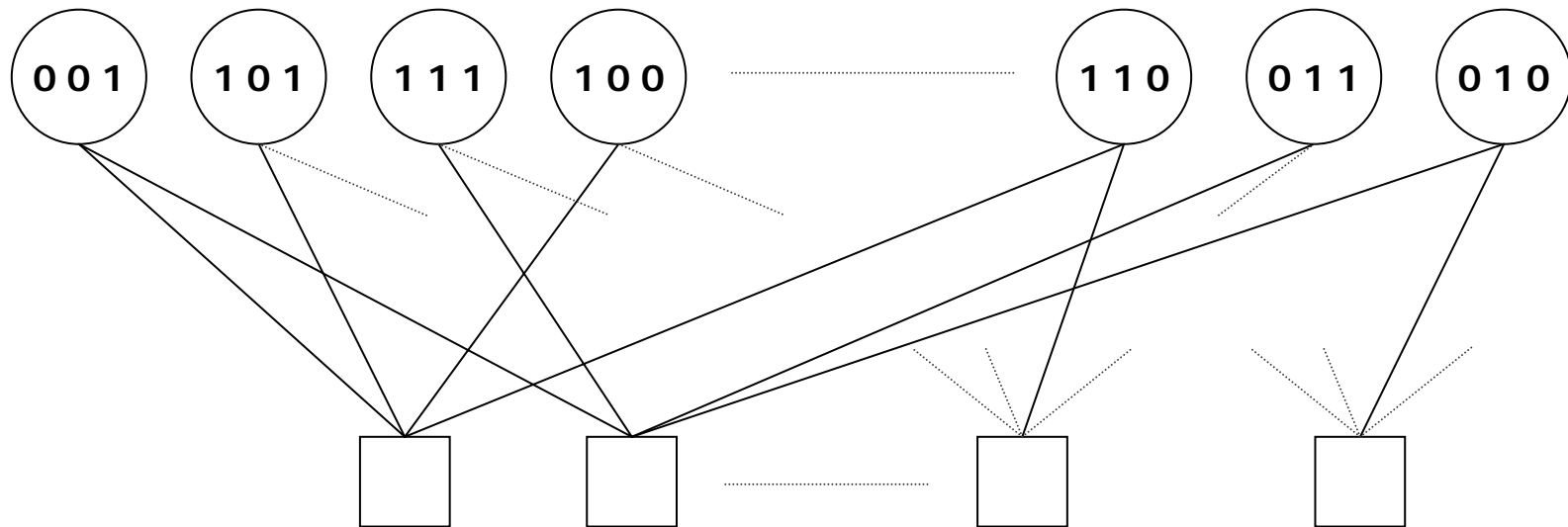
Example: alphabet $GF(8) = \{0, 1, 2, 3, 4, 5, 6, 7\}$



Advantages and related topics

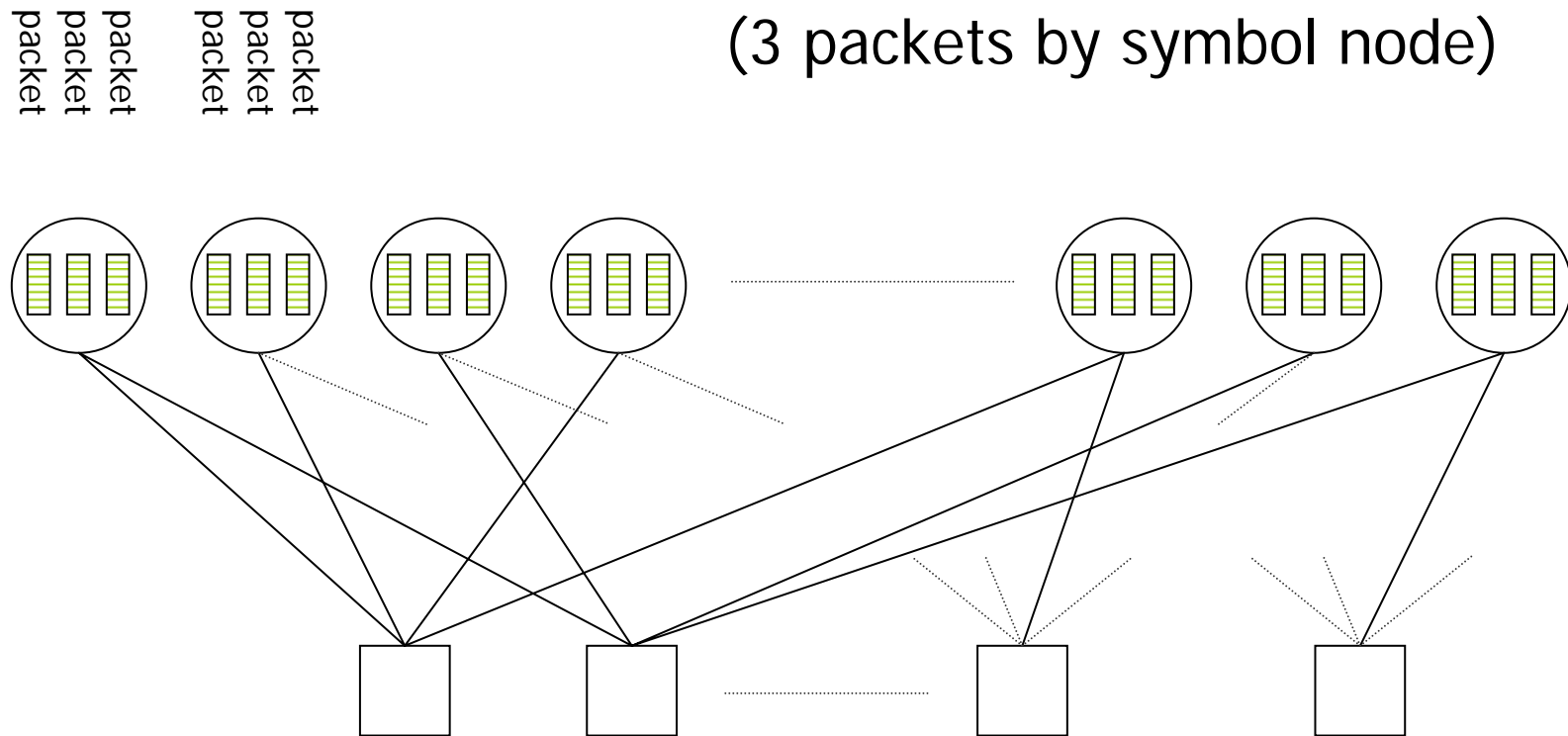
1. BLTE decoder can handle packets instead of bits

→ suited for upper-layer applications



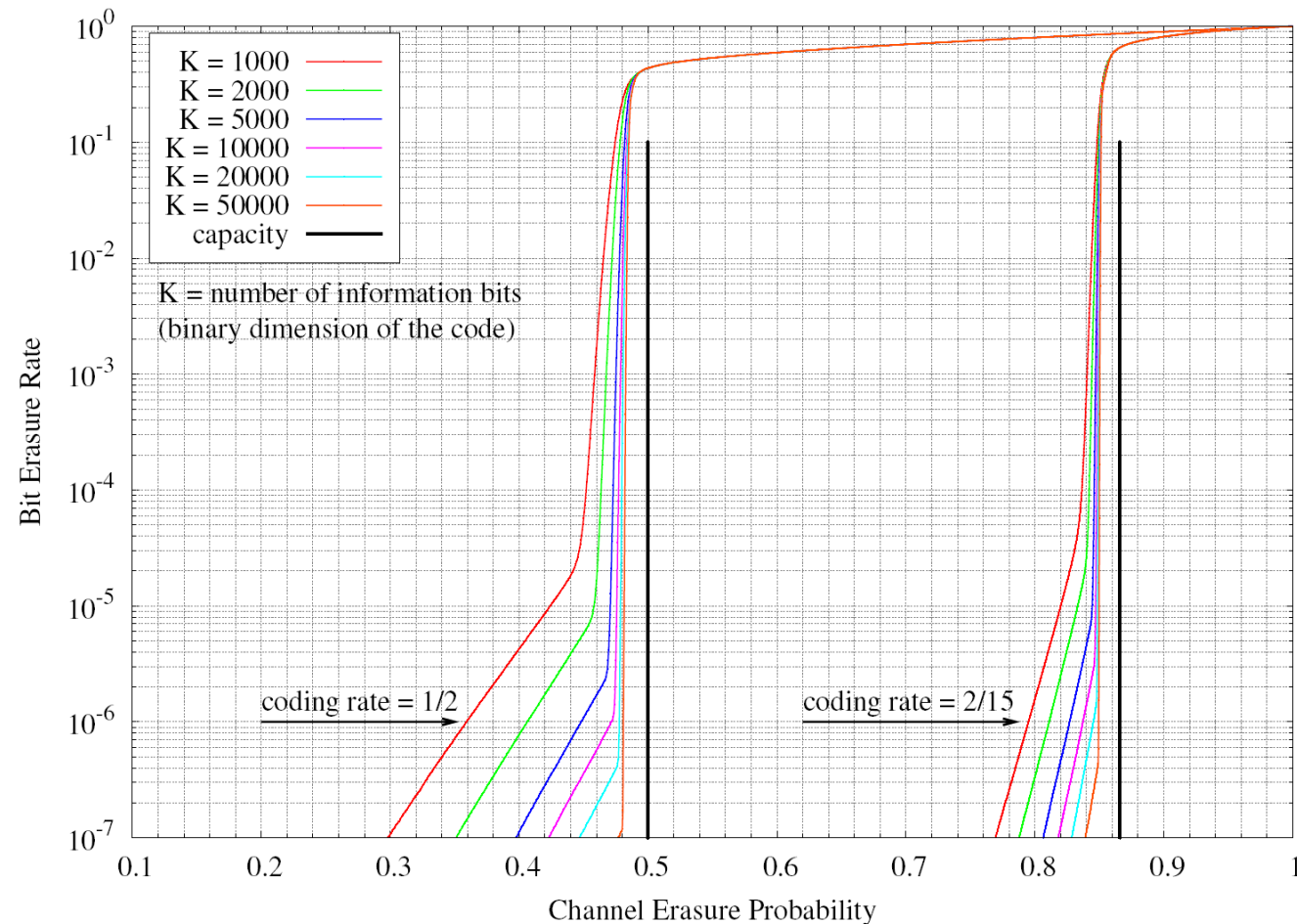
1. BLTE decoder can handle packets instead of bits

→ suited for upper-layer applications



2. Incremental redundancy

- transmit all the bits of the extended binary image
- cope with severe channel conditions



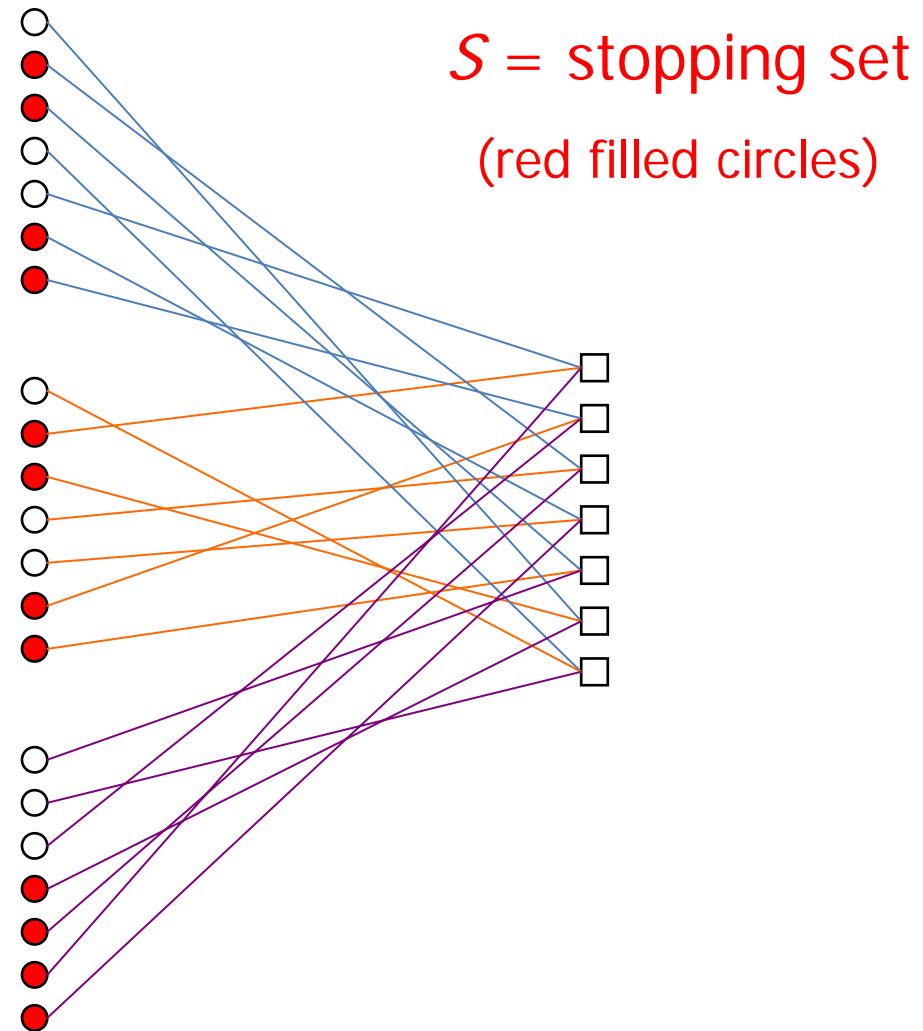
GF(16)

3. Stopping sets

S subset of extended bit-nodes, such that

- check-nodes that are neighbors of S are connected to S at least twice
- the complementary of S is an affine subspace
(if $a_{i \wedge j} \in S$, then either $a_i \in S$, or $a_j \in S$)

3. Stopping sets



4. Cycles in the extended binary matrix

(e_1, e_2, \dots, e_t) = cycle in the non-binary graph

(h_1, h_2, \dots, h_t) = corresponding edge labels

$h = h_1 h_2^{-1} \dots h_t$ (we assume here edge labels are invertible!)

P_h = extended binary representation

= permutation matrix of size $q-1$

Cyclic decomposition: $P_h = (1, 3, 5, 7) \cdot (2, 4) \cdot (6)$ (example of cyclic decomposition)
 $q=8$

1 cycle of length $4t$, 1 cycle of length $2t$, 1 cycle of length t

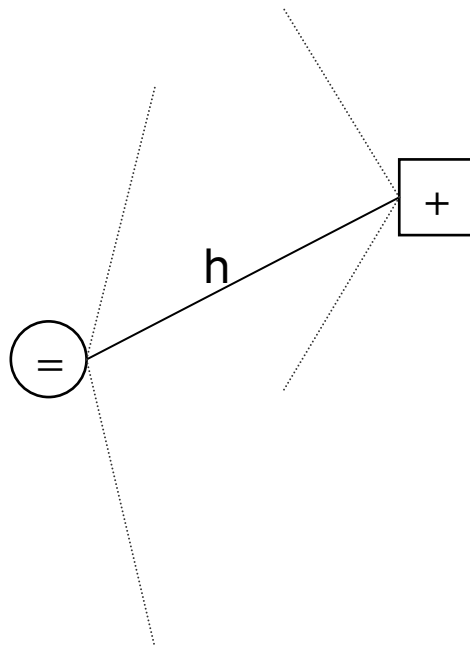
Non-binary matrix

$$\mathbf{H} = \begin{pmatrix} 1 & 3 & 4 & 6 & 0 & 0 \\ 0 & 2 & 0 & 7 & 1 & 5 \\ 5 & 0 & 1 & 0 & 2 & 4 \end{pmatrix}$$

Binary image

$$\mathbf{H}_{\text{bin}} = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ \hline 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}$$

Non-binary code



Extended binary image

