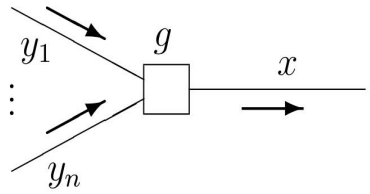
Message Passing (1)

Generic Edge:



Sum-Product Rule:

$$\mu_{g\to x}(x) \triangleq \sum_{y_1} \dots \sum_{y_n} g(x, y_1, \dots, y_n)\, \mu_{y_1\to g}(y_1) \cdots \mu_{y_n\to g}(y_n)$$

- Calculate probabilities

Max-Product Rule:

$$\mu_{g\to x}(x) \triangleq \max_{y_1} \dots \max_{y_n} g(x, y_1, \dots, y_n)\, \mu_{y_1\to g}(y_1) \cdots \mu_{y_n\to g}(y_n)$$

- Find entity with the highest prob.
- i.e. for decoding $\hat{x}(y) = argmax_{x\in C}\, p_{X|Y}(x|y)$

Exist other rules...

Message Passing (2)

Graphs with no cycles

- Pass all messages from the leafes to the root & back to the leafes

Graphs with cycles

- Iterative calculation
- Initialise all edges with a neutral message (i.e. with 1)
- Calculate all messages repeatedly with a specific schedule
- Stop conditions
  - Valid codeword is found
  - Available time is over

Normalize after each step, i.e. to $\sum \mu(.) = 1$

- Prevent message decay

FGs of linear parity check codes

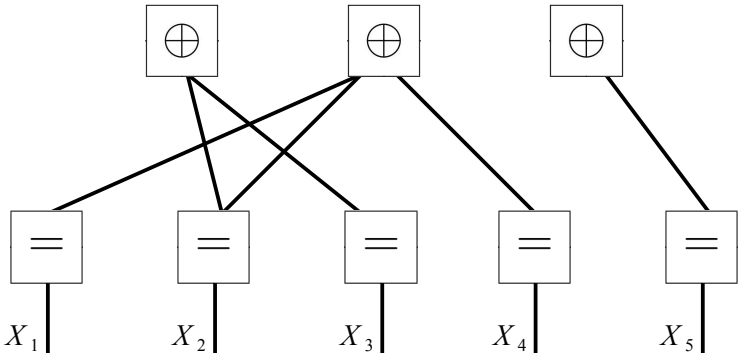
Used for error correction

Generator matrix G: $w = \boldsymbol{c} \cdot G$

Parity check matrix H: $H \cdot \boldsymbol{x} = \mathbf{0}, \forall\ \boldsymbol{x} \in C$

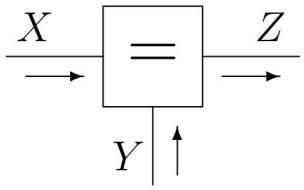
– H matrix for a (5,2) code: $H = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$

– Tanner graph of code:



Sum-product update rule (1)

Equality constraint node:



- Variable cloning
- Constraints: $X + Y - Z = 0$

$$f_{=}(x, y, z) \triangleq \delta(z - x) \cdot \delta(z - y)$$

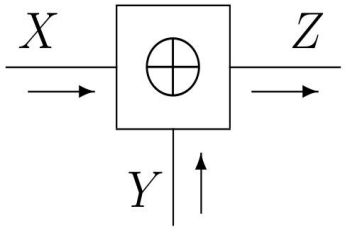
- Solution for a binary code:

$$\begin{pmatrix} \mu_Z(0) \\ \mu_Z(1) \end{pmatrix} = \begin{pmatrix} \mu_X(0) \cdot \mu_Y(0) \\ \mu_X(1) \cdot \mu_Y(1) \end{pmatrix}$$

- Derivation on example page

Sum-product update rule (2)

Even parity node:



- XOR function
- Constraint: $X, Y, Z \in \mathbb{F}_2$:

$$f_{\oplus}(x, y, z) \triangleq \delta(x \oplus y \oplus z)$$

- Solution for binary code:

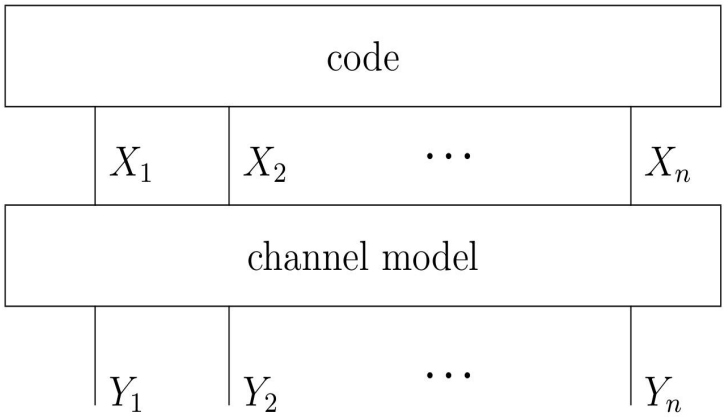
$$\begin{pmatrix} \mu_Z(0) \\ \mu_Z(1) \end{pmatrix} = \begin{pmatrix} \mu_X(0) \cdot \mu_Y(0) + \mu_X(1) \cdot \mu_Y(1) \\ \mu_X(1) \cdot \mu_Y(0) + \mu_X(0) \cdot \mu_Y(1) \end{pmatrix}$$

- Derivation on example page

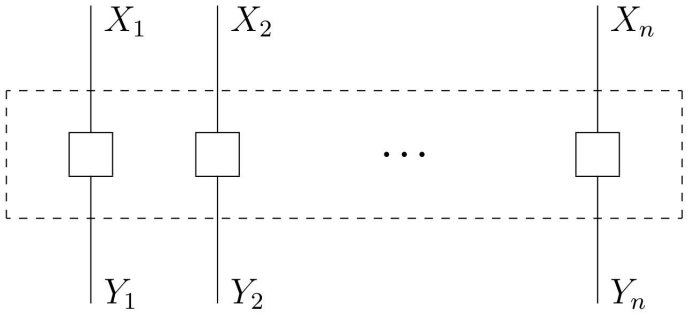
FFG on codes

Combine code and channel

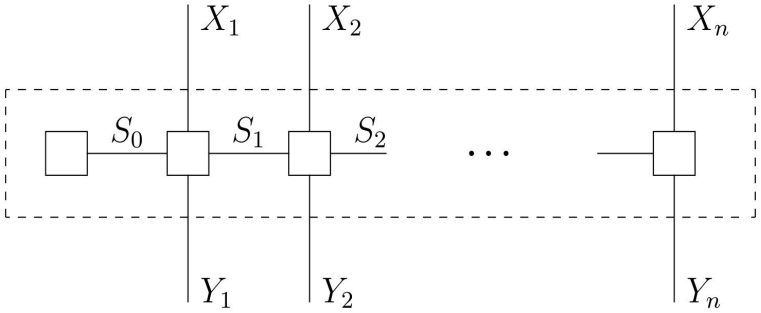
– Code/channel model



– Memoryless channel

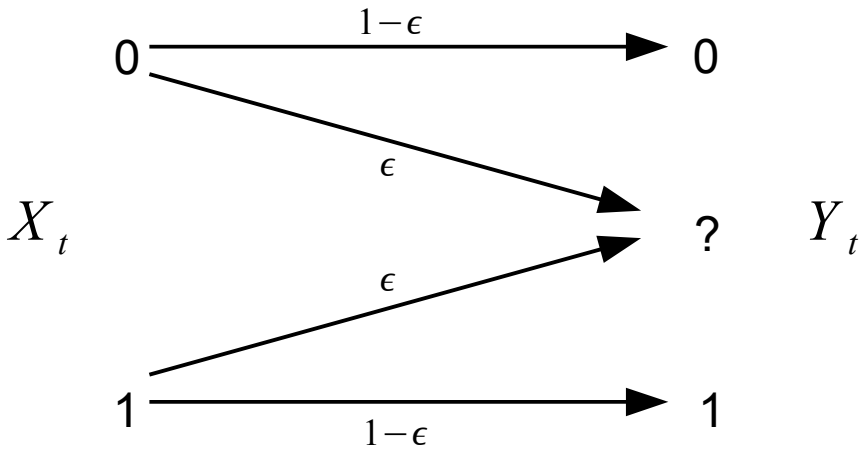


– State space model



Binary Erasure Channel (BEC)

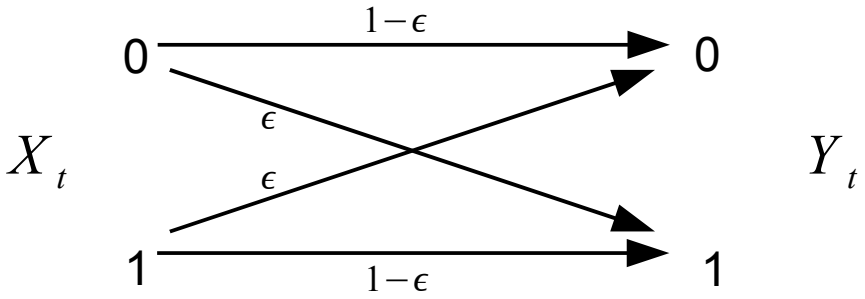
Channel Model:



- Alphabeth: $X_t \in \{0,1\}$ $Y_t \in \{0,1,?\}$
- Binary channel
- Memoryless Model

Binary Symmetric Channel (BSC)

Channel Model:



- Alphabeth: $X_t \in \{0,1\}$

  $Y_t \in \{0,1\}$

- Binary channel
- Memoryless model
- Bit-flip model, i.e. in a RAM

Sum-product algorithm: example (1)

Sum-Product (Belief Propagation) Algorithm: An Example from [1]:

- Linear Binary Code:
  C = { (0,0,0,0), (0,1,1,1), (1,0,1,1), (1,1,0,0) }
- Binary Symmetric channel:
  $\epsilon = 0.1$
- Received Message:
  $(Y_{1,} Y_{2,} Y_{3,} Y_4) = (0,0,1,0)$
- Posteriory probability:
  $p(x_i | y_1 \ldots y_4) = ?, i = 1 \ldots 4$
- Calculation on example page

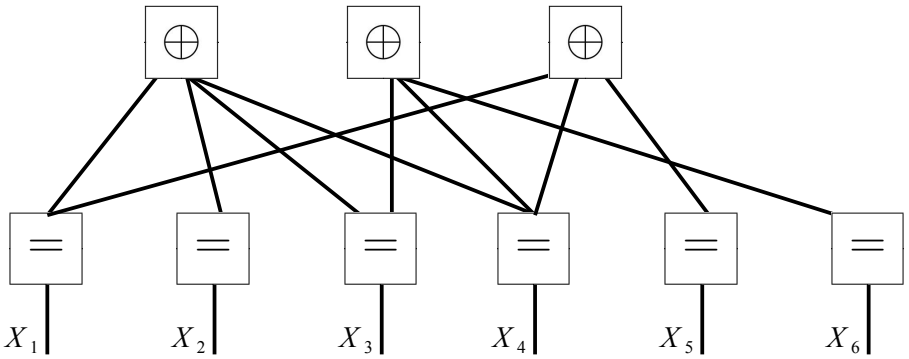
Sum-product algorithm: example (2)

Matlab demo: LDPC fragment decoding

- Iterative Sum-Product algorithm
- Matlab library: SumProductLab for Factor Graphs [4]
- Parity check matrix:

$$H = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

- Factor graph:



- BSC, $\epsilon = 0.1$
- Posteriori probabilities: $p(x_i | y_1 \dots y_6) = ?, i = 1 \dots 6$

Discussion

References

[1] H.A. Loeliger, “An Introduction to Factor Graphs”, IEEE Signal Processing Mag., Jan. 2004.

[2] T. Richardson, R. Urbanke, “Modern Coding Theory”, Cambridge University Press, 2008.

[3] H.A. Loeliger et. al., “The Factor Graph Approach to Model-Based Signal Processing” in Proc. 2007 IEEE, Jun. 2007.

[4] Henry Leung, “SumProductLab for Factor Graphs”, Matlab Central, http://www.mathworks.com/matlabcentral/fileexchange/26607-sumproductlab-for-factor-graphs, 2010.