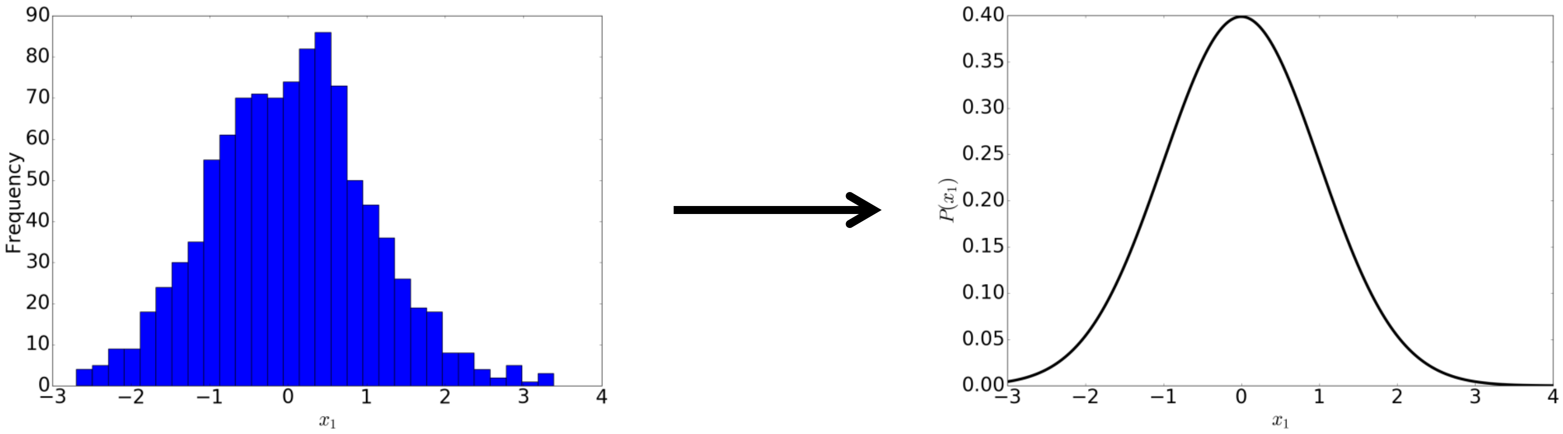
Belief Propagation

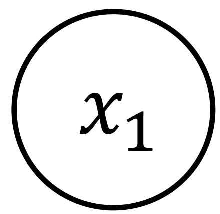
Algorithm Interest Group
presentation by Eli Chertkov

Inference

Statistical inference is the determination of an underlying probability distribution from observed data.



Probabilistic Graphical Models



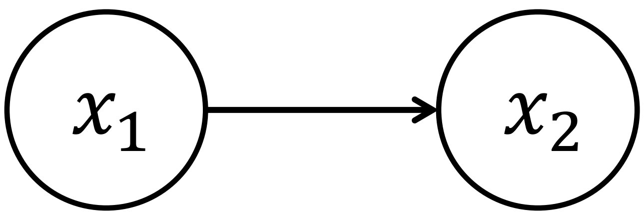
$$P(x_1) = \phi_1(x_1)$$

Probabilistic Graphical Models

$$P(x_1, x_2) = \phi_1(x_1)\phi_2(x_2)$$

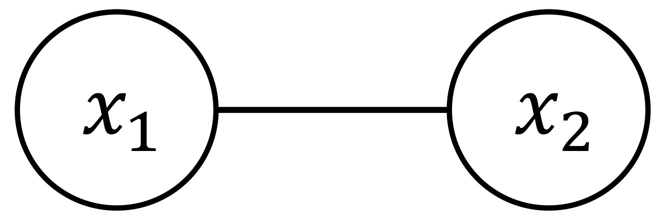
Probabilistic Graphical Models

Directed: Bayesian Network



$$P(x_1, x_2) = P(x_2|x_1)\phi_1(x_1)$$

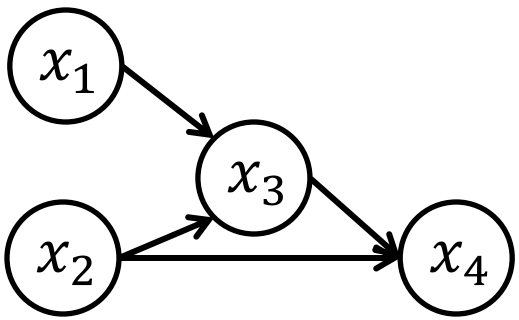
Undirected: Markov Random Field



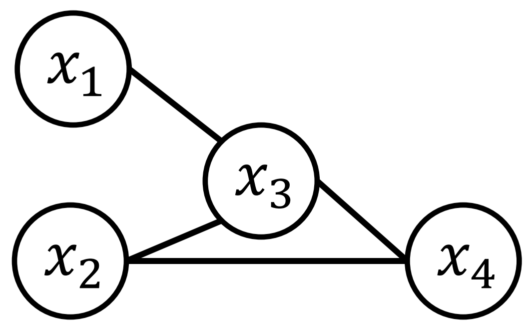
$$P(x_1, x_2) = \phi_{12}(x_1, x_2)$$

Probabilistic Graphical Models

Directed: Bayesian Network **Undirected: Markov Random Field**



$$P(x_1, x_2, x_3, x_4) = P(x_4|x_3, x_2)P(x_3|x_2, x_1)\phi_2(x_2)\phi_1(x_1)$$

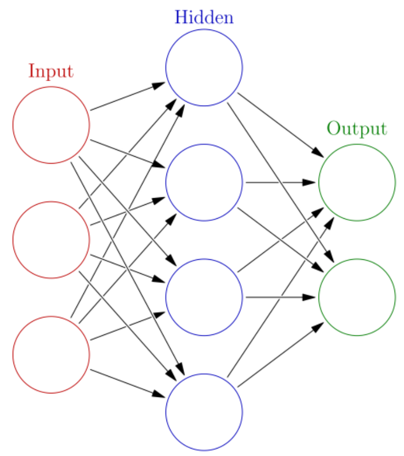


$$P(x_1, x_2, x_3, x_4) = \phi_{43}(x_4, x_3)\phi_{42}(x_4, x_2)\phi_{32}(x_3, x_2)\phi_{31}(x_3, x_1)\phi_2(x_2)\phi_1(x_1)$$

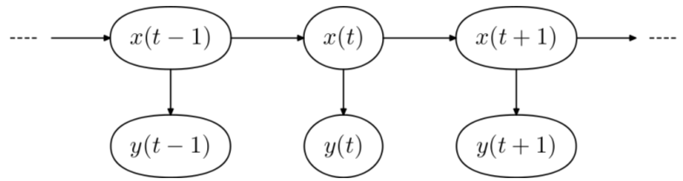
Probabilistic Graphical Models

Directed: Bayesian Network

Artificial Neural Network
(Deep Learning)

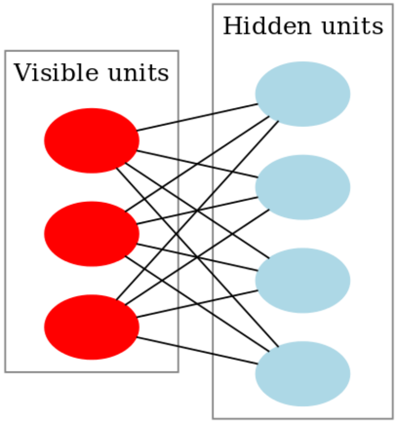


Hidden Markov Model

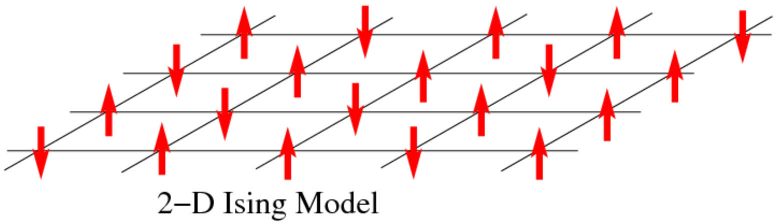


Undirected: Markov Random Field

Restricted Boltzmann Machine



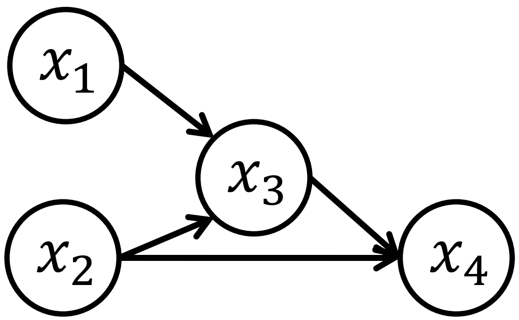
Ising Model



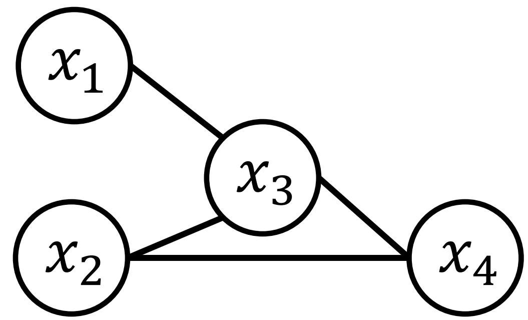
Source: Wikipedia

Factor Graphs

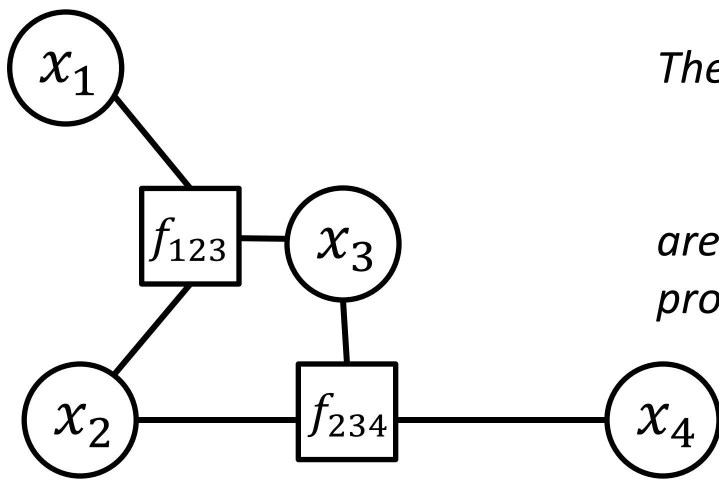
Directed: Bayesian Network



Undirected: Markov Random Field



These probability distributions can both be represented in terms of **factor graphs**



The factors

$$f_{123} = f_{123}(x_1, x_2, x_3)$$
$$f_{234} = f_{234}(x_2, x_3, x_4)$$

are chosen to match the original probability distributions.

$$P(x_1, x_2, x_3, x_4) = f_{123}(x_1, x_2, x_3) f_{234}(x_2, x_3, x_4)$$

Belief Propagation Outline

- The goal of BP is to compute the marginal probability distribution for a random variable x_i in a graphical model:

$$P(x_i) = \sum_{\{x_j\} \setminus x_i} P(x_1, \dots, x_N)$$

- The probability distribution of a graphical model can be represented as a factor graph so that

$$P(x_i) = \sum_{\{x_j\} \setminus x_i} \prod_{f \in ne(x_i)} f(x_i, \{x_j\}_f)$$

where $\{x_j\}_f$ is the subset of the variables involved in factor f.

- By interchanging the product and sum, we can write

$$P(x_i) = \prod_{f \in ne(x_i)} \mu_{f \to x_i}(x_i)$$

where $\mu_{f \to x_i}(x_i) = \sum_{\{x_j\}_f} f(x_i, \{x_j\}_f)$ is called a **message**.

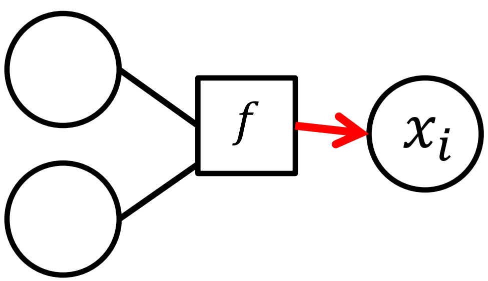
Belief Propagation Message Passing

BP is a message-passing algorithm. The idea is to pass information through your factor graph by locally updating the messages between nodes.

Once the messages have converged, then you can efficiently evaluate the marginal distribution for each variable:

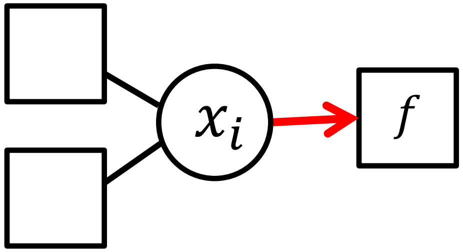
$$P(x_i) = \prod_{f \in ne(x_i)} \mu_{f \to x_i}(x_i)$$

There are two types of message updates:



Factor node to variable node

$$\mu_{f \to x_i}(x_i) = \sum_{\{x_j \in ne(f) \backslash x_i\}} f(\{x_j\}, x_i) \prod_{x_j} \mu_{x_j \to f}(x_j)$$

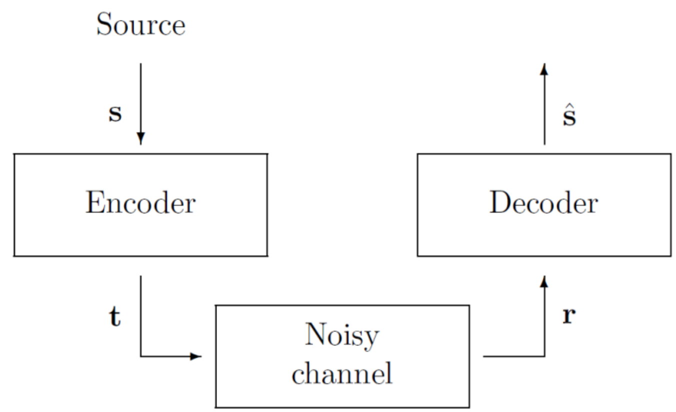


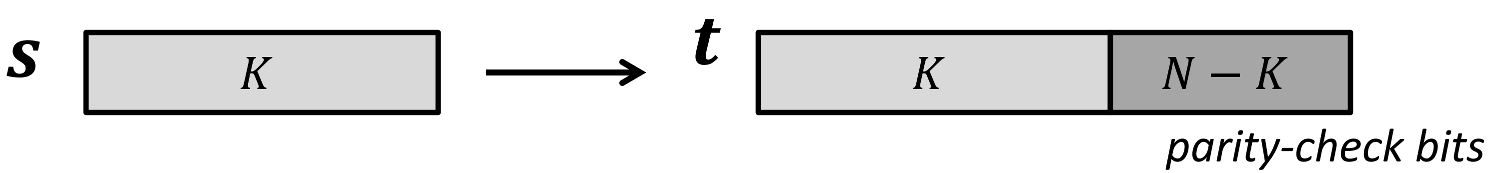
Variable node to factor node

$$\mu_{x_i \to f}(x_i) = \prod_{f' \in ne(x_i) \backslash f} \mu_{f' \to x_i}(x_i)$$

Killer app: Error-correcting codes

To prevent the degradation of a binary signal through a noisy channel, we ***encode*** our original signal **s** into a redundant one **t**.





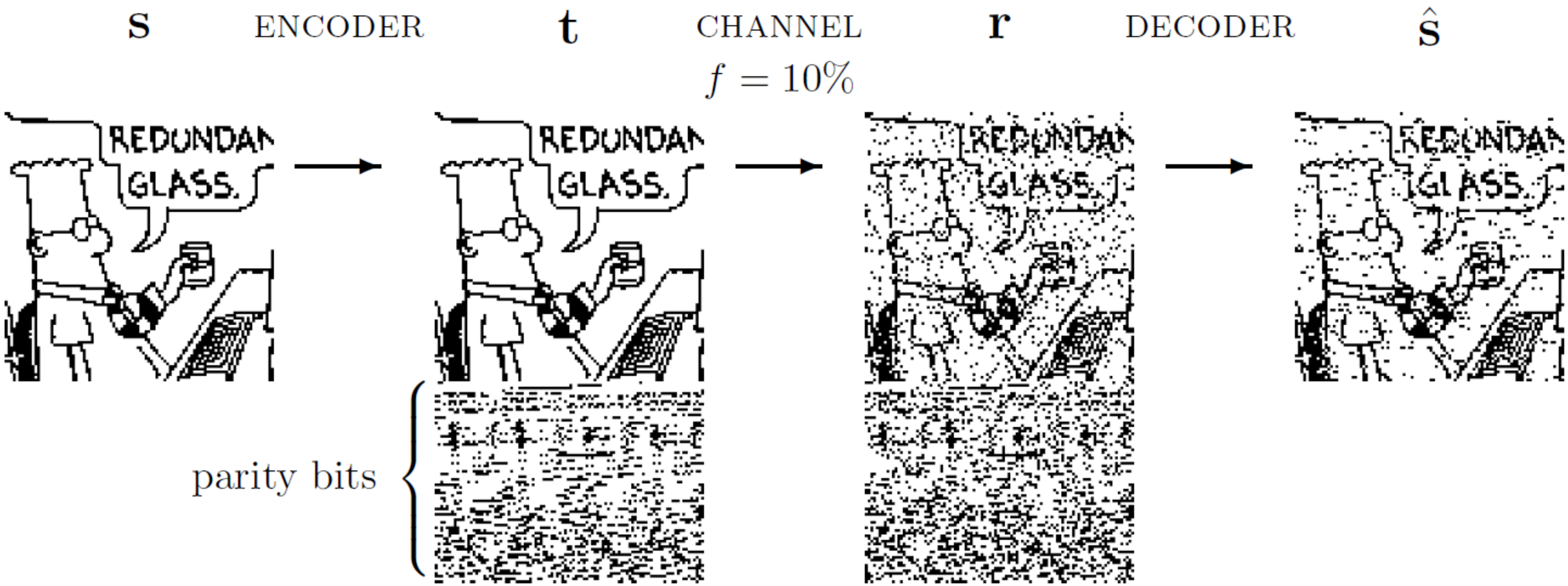
A theoretically useful encoding scheme is linear block coding, which relates the two signals by a (binary) linear transformation

$$\boldsymbol{t} = \boldsymbol{G}^T \boldsymbol{s}$$

When the matrix $\boldsymbol{G}^T$ is random and sparse, the encoding is called a *low-density parity check* (LDPC) code.

Decoding the degraded signal **r** of a LDPC code, i.e., inferring the original signal **s**, is an NP-complete problem. Nonetheless, BP is efficient at providing an approximate solution.

Linear block code visualization



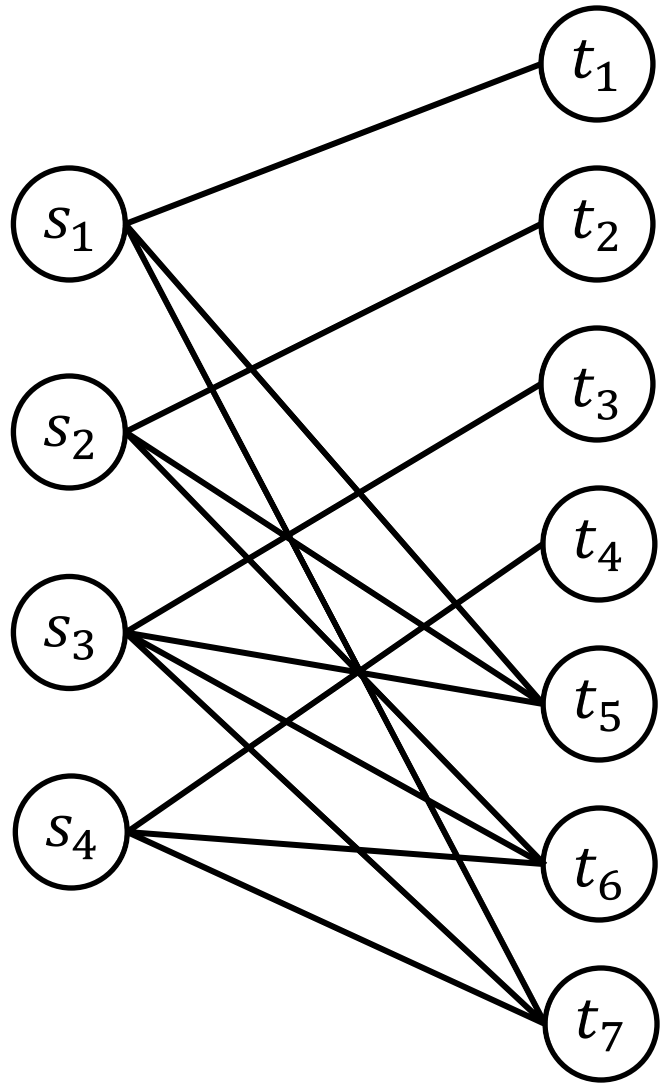
Source: Information Theory, Inference, and Learning Algorithms

Linear block code as a graphical model

$$\boldsymbol{t} = \boldsymbol{G}^T \boldsymbol{s}$$

$$\boldsymbol{G}^T = \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \\ 1 & 1 & 1 & \\ & 1 & 1 & 1 \\ 1 & & 1 & 1 \end{pmatrix}$$

$P(s_1, s_2, s_3, s_4, t_1, \dots, t_7) \rightarrow$



$s_i, t_j \in \{0,1\}$ are binary random variables

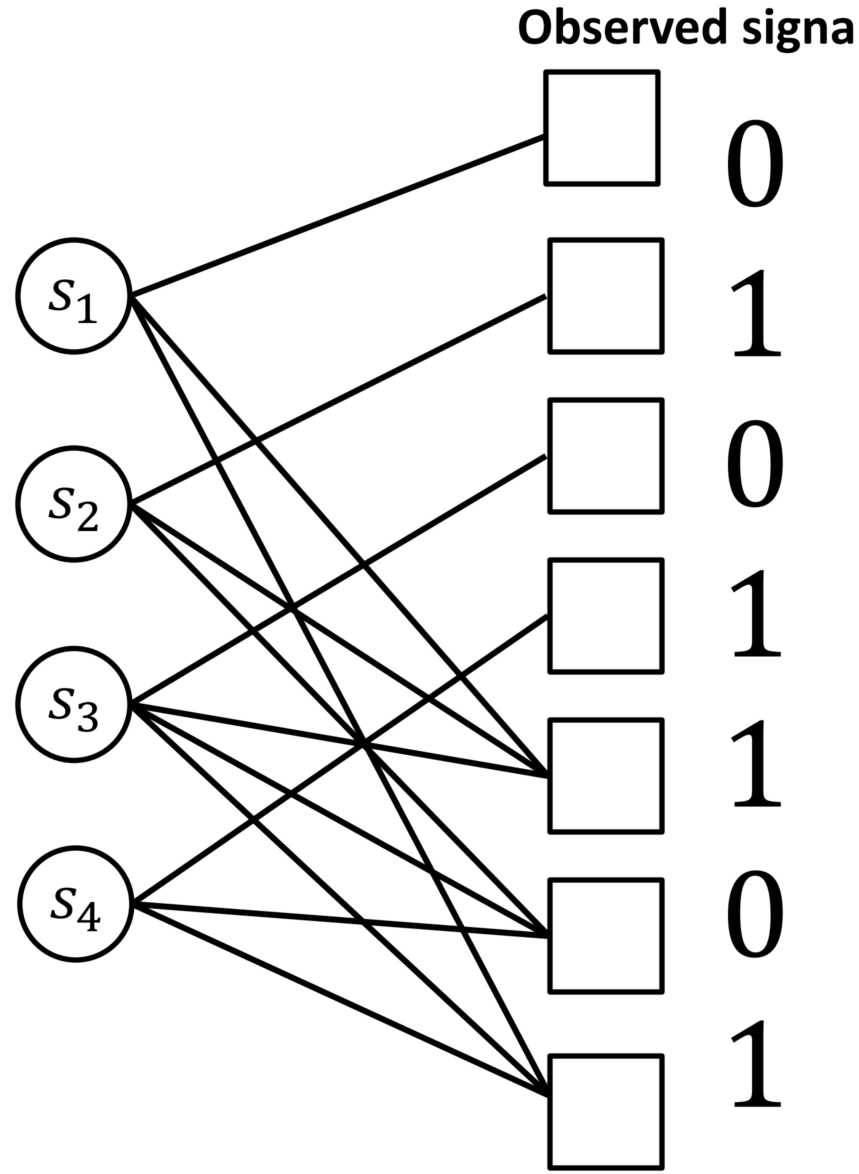
Linear block code as a graphical model

When decoding a signal, we observe the transmitted bits t_j and try to find the most likely source bits s_i.

This means we want to maximize

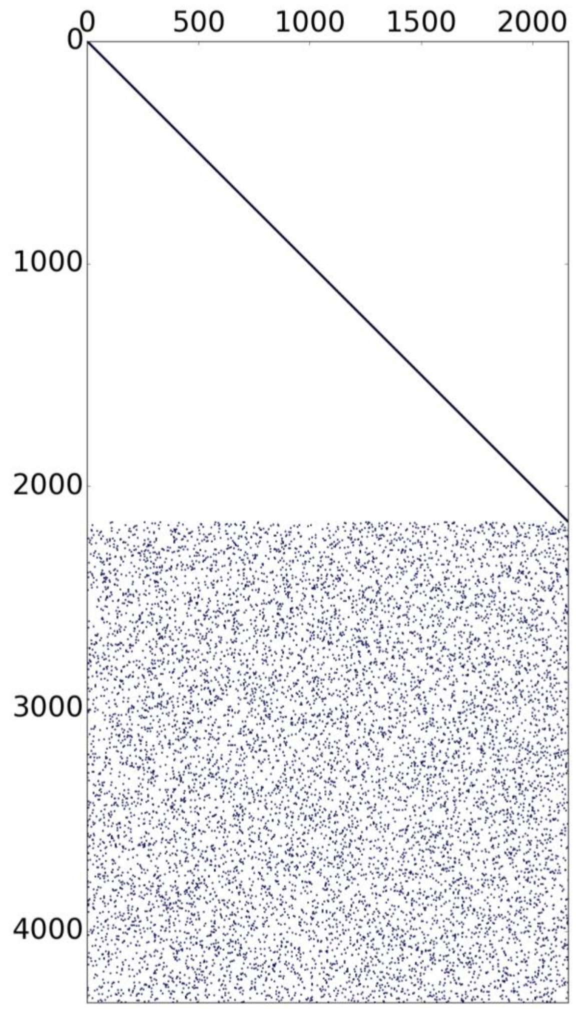
$$P(s_1, s_2, s_3, s_4 | t_1, \ldots, t_7 = 0101101)$$

Belief Propagation is an efficient way to compute the marginal probability distribution $P(s_i)$ of the source bits s_i.

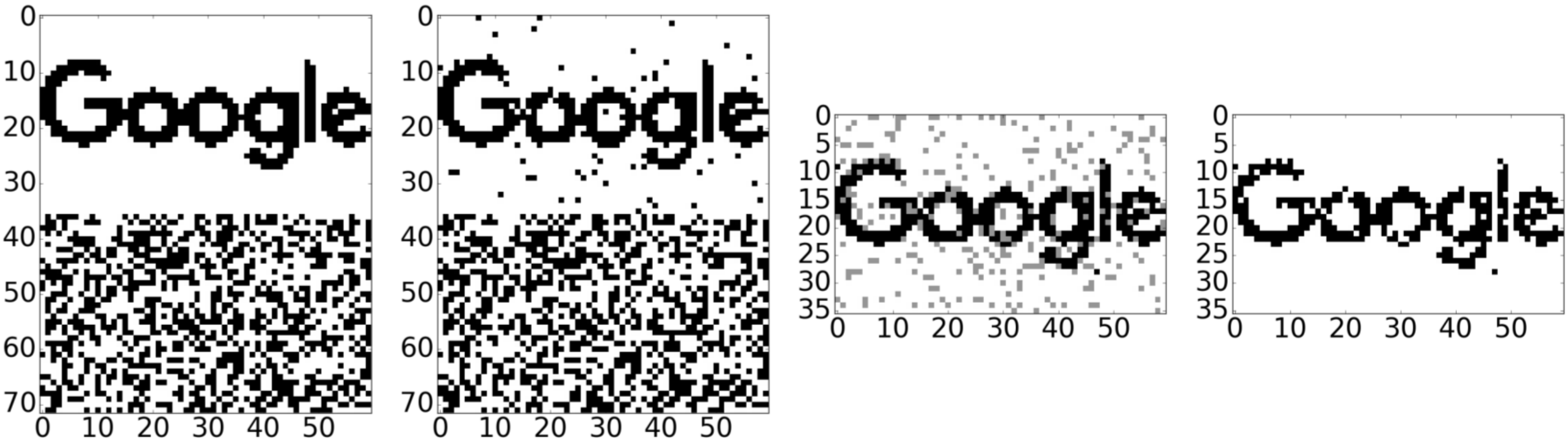


My toy LDPC decoding example

Encoding matrix $= \boldsymbol{G}^{\boldsymbol{T}} =$



My toy LDPC decoding example



Encoded signal **Noisy transmitted signal** **Marginal probabilities** **Reconstructed signal**

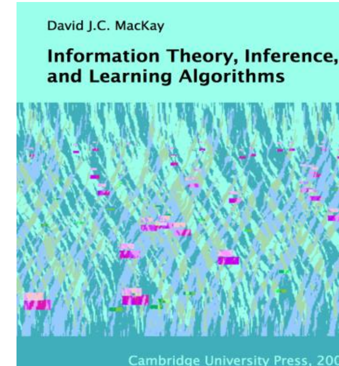
Note: There is a very similar message-passing algorithm, called the max-product (or min-sum, or Viterbi) algorithm, which computes the maximum probability configuration of the probability distribution $\mathrm{x}^* = \mathrm{argmax}_{\mathrm{x}}\, \mathrm{P}(\mathrm{x})$, which might be better suited for this decoding task.

References

Information Theory, Inference, and Learning Algorithms by David MacKay.

Yedidia, J.S.; Freeman, W.T.; Weiss, Y., "Understanding Belief Propagation and Its Generalizations", *Exploring Artificial Intelligence in the New Millennium* (2003) Chap. 8, pp. 239-269.

Pattern Recognition and Machine Learning by Christopher Bishop.




Understanding Belief Propagation and its Generalizations

Jonathan S. Yedidia, William T. Freeman, and Yair Weiss

TR2001-22 November 2001

Abstract