ECEN 655: Advanced Channel Coding

Lecture: Marginalization and Factor Graphs

Henry D. Pfister

Department of Electrical and Computer Engineering
Texas A&M University

Outline

The Importance of Marginalization (1)

- Consider a random vector $(X_1, X_2, \ldots, X_6) \in \mathcal{X}^6$ where
  - the vector is indirectly observed via the random variable Y
  - the posterior probability given $Y = y$ satisfies:

$$\mathbb{P}(X_1 = x_1, X_2 = x_2, \ldots, X_6 = x_6 | Y = y) \propto f(x_1, x_2, x_3, x_4, x_5, x_6)$$

The Importance of Marginalization (2)

- Marginalizing all variables except X_1 gives

$$\mathbb{P}(X_1 = x_1 | Y = y) \propto \sum_{(x_2,\ldots,x_6)\in\mathcal{X}^5} f(x_1, x_2, x_3, x_4, x_5, x_6)$$

- Thus, the symbol MAP decision for X_1 is given by

$$\hat{x}_1 = \arg\max_{x_1\in\mathcal{X}} \sum_{(x_2,\ldots,x_6)\in\mathcal{X}^5} f(x_1, x_2, x_3, x_4, x_5, x_6)$$

- For many problems, the computing the marginalization sum is a significant challenge

The Importance of Factorization (1)

- Simply summing requires $|\mathcal{X}|^5$ operations for each $x_1 \in \mathcal{X}$:

$$g_1(x_1) \triangleq \sum_{x_2^6 \in \mathcal{X}^5} f(x_1, x_2, x_3, x_4, x_5, x_6)$$

  - Thus, we need $|\mathcal{X}|^6$ operations

- If f factors as follows, then marginalization can be simplified:

$$f(x_1, x_2, x_3, x_4, x_5, x_6) = f_1(x_1, x_2, x_3) f_2(x_1, x_4, x_6) f_3(x_4) f_4(x_4, x_5)$$

The Importance of Factorization (2)

For example, we can write $g_1(x_1)$ as:

$$= \sum_{x_2^6} f_1(x_1, x_2, x_3) f_2(x_1, x_4, x_6) f_3(x_4) f_4(x_4, x_5)$$

The Importance of Factorization (2)

For example, we can write $g_1(x_1)$ as:

$$= \sum_{x_2^6} f_1(x_1, x_2, x_3) f_2(x_1, x_4, x_6) f_3(x_4) f_4(x_4, x_5)$$

$$= \sum_{x_2^5} f_1(x_1, x_2, x_3) f_3(x_4) f_4(x_4, x_5) \left[\sum_{x_6} f_2(x_1, x_4, x_6) \right]$$

The Importance of Factorization (2)

For example, we can write $g_1(x_1)$ as:

$$= \sum_{x_2^6} f_1(x_1, x_2, x_3) f_2(x_1, x_4, x_6) f_3(x_4) f_4(x_4, x_5)$$

$$= \sum_{x_2^5} f_1(x_1, x_2, x_3) f_3(x_4) f_4(x_4, x_5) \left[\sum_{x_6} f_2(x_1, x_4, x_6) \right]$$

$$= \sum_{x_2^4} f_1(x_1, x_2, x_3) f_3(x_4) \left[\sum_{x_5} f_4(x_4, x_5) \right] \left[\sum_{x_6} f_2(x_1, x_4, x_6) \right]$$

The Importance of Factorization (2)

For example, we can write $g_1(x_1)$ as:

$$= \sum_{x_2^6} f_1(x_1, x_2, x_3) f_2(x_1, x_4, x_6) f_3(x_4) f_4(x_4, x_5)$$

$$= \sum_{x_2^5} f_1(x_1, x_2, x_3) f_3(x_4) f_4(x_4, x_5) \left[\sum_{x_6} f_2(x_1, x_4, x_6) \right]$$

$$= \sum_{x_2^4} f_1(x_1, x_2, x_3) f_3(x_4) \left[\sum_{x_5} f_4(x_4, x_5) \right] \left[\sum_{x_6} f_2(x_1, x_4, x_6) \right]$$

$$= \sum_{x_2^3} f_1(x_1, x_2, x_3) \left[\sum_{x_4} f_3(x_4) \left[\sum_{x_5} f_4(x_4, x_5) \right] \left[\sum_{x_6} f_2(x_1, x_4, x_6) \right] \right]$$

The Importance of Factorization (2)

For example, we can write $g_1(x_1)$ as:

$$= \sum_{x_2^6} f_1(x_1,x_2,x_3) f_2(x_1,x_4,x_6) f_3(x_4) f_4(x_4,x_5)$$

$$= \sum_{x_2^5} f_1(x_1,x_2,x_3) f_3(x_4) f_4(x_4,x_5) \left[\sum_{x_6} f_2(x_1,x_4,x_6) \right]$$

$$= \sum_{x_2^4} f_1(x_1,x_2,x_3) f_3(x_4) \left[\sum_{x_5} f_4(x_4,x_5) \right] \left[\sum_{x_6} f_2(x_1,x_4,x_6) \right]$$

$$= \sum_{x_2^3} f_1(x_1,x_2,x_3) \left[\sum_{x_4} f_3(x_4) \left[\sum_{x_5} f_4(x_4,x_5) \right] \left[\sum_{x_6} f_2(x_1,x_4,x_6) \right] \right]$$

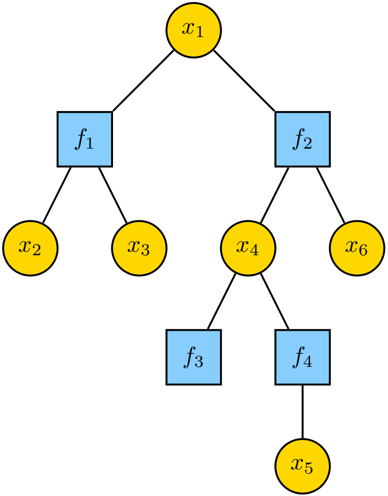
$$= \sum_{x_2} \left[\sum_{x_3} f_1(x_1,x_2,x_3) \right] \left[\sum_{x_4} f_3(x_4) \left[\sum_{x_5} f_4(x_4,x_5) \right] \left[\sum_{x_6} f_2(x_1,x_4,x_6) \right] \right]$$

This implementation requires roughly $2\,|\mathcal{X}|^3 + 5\,|\mathcal{X}|^2$ operations

The Factor Graph and Leaf Removal

Graphical Marginalization:

- Factor graph G shows variables involved in each factor
- If the factor graph is a tree, then efficient marginalization is associated with leaf removal
- After marginalization of a leaf node variable, the node is removed and its contribution is absorbed into the factor node.
- This costs $|\mathcal{X}|^d$, where d is the degree of updated factor node

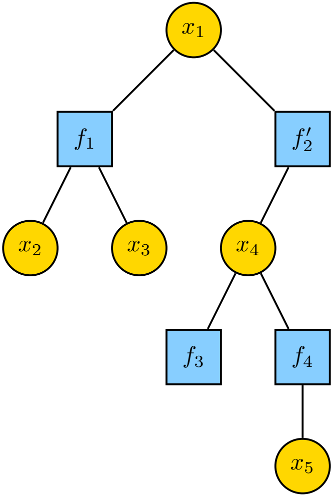


$$g_1(x_1) = \sum_{x_2^5} f_1(x_1, x_2, x_3) f_3(x_4) f_4(x_4, x_5) \left[\sum_{x_6} f_2(x_1, x_4, x_6) \right]$$

The Factor Graph and Leaf Removal

Graphical Marginalization:

- Factor graph G shows variables involved in each factor
- If the factor graph is a tree, then efficient marginalization is associated with leaf removal
- After marginalization of a leaf node variable, the node is removed and its contribution is absorbed into the factor node.
- This costs $|\mathcal{X}|^d$, where d is the degree of updated factor node

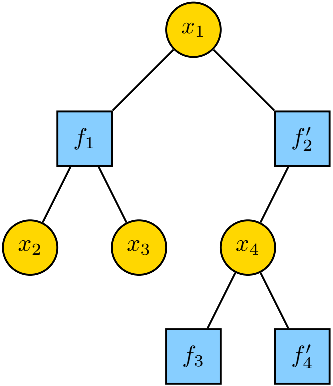


$$g_1(x_1) = \sum_{x_2^4} f_1(x_1,x_2,x_3) f_3(x_4) \left[\sum_{x_5} f_4(x_4,x_5) \right] f_2'(x_1,x_4)$$

The Factor Graph and Leaf Removal

Graphical Marginalization:

- Factor graph G shows variables involved in each factor
- If the factor graph is a tree, then efficient marginalization is associated with leaf removal
- After marginalization of a leaf node variable, the node is removed and its contribution is absorbed into the factor node.
- This costs $|\mathcal{X}|^d$, where d is the degree of updated factor node

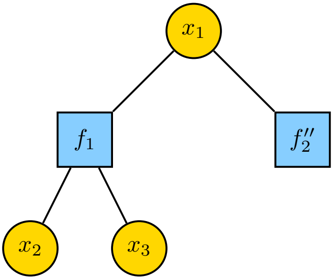


$$g_1(x_1) = \sum_{x_2^3} f_1(x_1, x_2, x_3) \left[\sum_{x_4} f_3(x_4) f_4'(x_4) f_2'(x_1, x_4) \right]$$

The Factor Graph and Leaf Removal

Graphical Marginalization:

- Factor graph G shows variables involved in each factor
- If the factor graph is a tree, then efficient marginalization is associated with leaf removal
- After marginalization of a leaf node variable, the node is removed and its contribution is absorbed into the factor node.
- This costs $|\mathcal{X}|^d$, where d is the degree of updated factor node

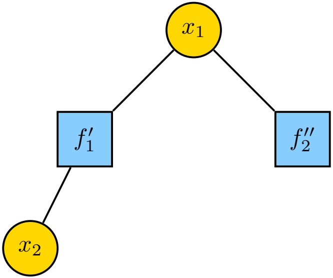


$$g_1(x_1) = \sum_{x_2} \left[\sum_{x_3} f_1(x_1, x_2, x_3) \right] f_2''(x_1)$$

The Factor Graph and Leaf Removal

Graphical Marginalization:

- Factor graph G shows variables involved in each factor
- If the factor graph is a tree, then efficient marginalization is associated with leaf removal
- After marginalization of a leaf node variable, the node is removed and its contribution is absorbed into the factor node.
- This costs $|\mathcal{X}|^d$, where d is the degree of updated factor node

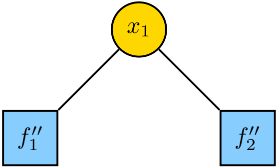


$$g_1(x_1) = \left[\sum_{x_2} f_1'(x_1, x_2) \right] f_2''(x_1)$$

The Factor Graph and Leaf Removal

Graphical Marginalization:

- Factor graph G shows variables involved in each factor
- If the factor graph is a tree, then efficient marginalization is associated with leaf removal
- After marginalization of a leaf node variable, the node is removed and its contribution is absorbed into the factor node.
- This costs $|\mathcal{X}|^d$, where d is the degree of updated factor node



$$g_1(x_1) = f_1''(x_1) f_2''(x_1)$$

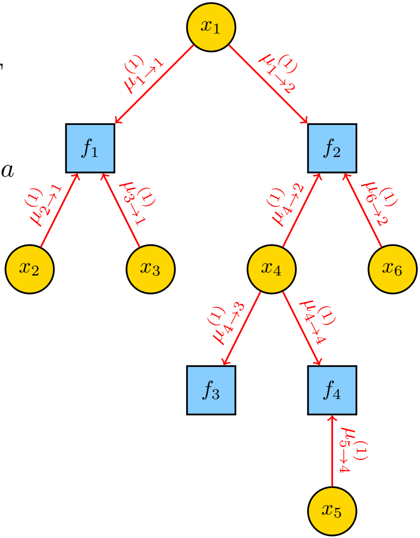
Marginalization via Message Passing: Definition

Factor Graph (FG):

- Variable nodes V, Factor nodes F
- Edges: $(i, a) \in E \subseteq V \times F$
- $F(i)/V(a) =$ neighbors of node-i/a

Message Passing (MP):

- Init: $\mu_{i \to a}^{(1)}(x_i) = 1 \; \forall \; (i, a) \in E$



Marginalization via Message Passing: Definition

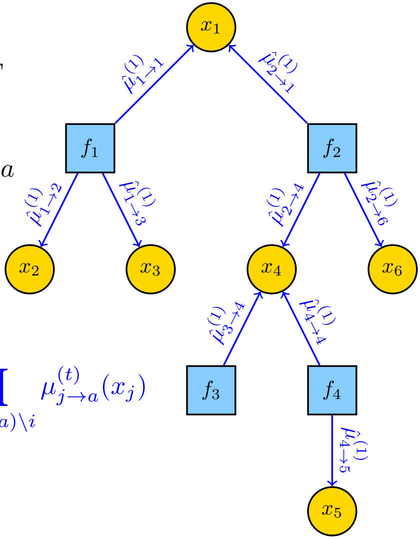
Factor Graph (FG):

- Variable nodes V, Factor nodes F
- Edges: $(i,a) \in E \subseteq V \times F$
- $F(i)/V(a) =$ neighbors of node-i/a

Message Passing (MP):

- Init: $\mu^{(1)}_{i\to a}(x_i) = 1 \ \forall\ (i,a) \in E$
- factor-a to variable-i message:

$$\hat{\mu}^{(t)}_{a\to i}(x_i) = \sum_{x_{V(a)\setminus i}} f_a(x_{V(a)}) \prod_{j\in V(a)\setminus i} \mu^{(t)}_{j\to a}(x_j)$$



Marginalization via Message Passing: Definition

Factor Graph (FG):

- Variable nodes V, Factor nodes F
- Edges: $(i,a) \in E \subseteq V \times F$
- $F(i)/V(a) =$ neighbors of node-i/a

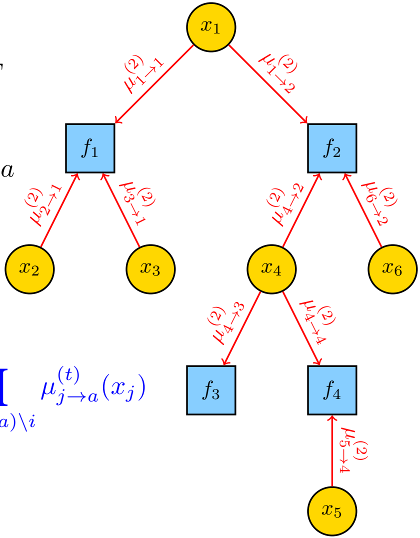
Message Passing (MP):

- Init: $\mu^{(1)}_{i\to a}(x_i) = 1 \ \forall\ (i,a) \in E$
- factor-a to variable-i message:

$$\hat{\mu}^{(t)}_{a\to i}(x_i) = \sum_{x_{V(a)\setminus i}} f_a(x_{V(a)}) \prod_{j\in V(a)\setminus i} \mu^{(t)}_{j\to a}(x_j)$$

- variable-i to factor-a message:

$$\mu^{(t+1)}_{i\to a}(x_i) = \prod_{b\in F(i)\setminus a} \hat{\mu}^{(t)}_{b\to i}(x_i)$$



Marginalization via Message Passing: Definition

Factor Graph (FG):

- Variable nodes V, Factor nodes F
- Edges: $(i,a) \in E \subseteq V \times F$
- $F(i)/V(a)$ = neighbors of node-i/a

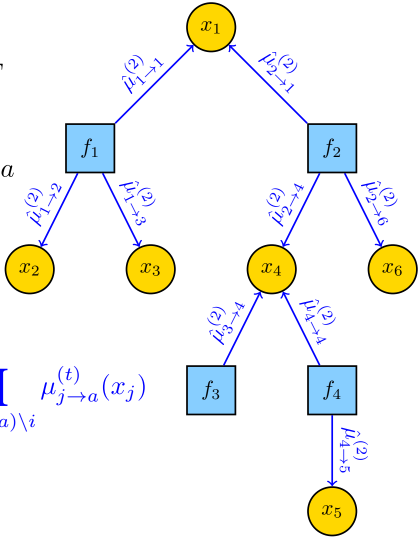
Message Passing (MP):

- Init: $\mu_{i\to a}^{(1)}(x_i) = 1 \ \forall \ (i,a) \in E$
- factor-a to variable-i message:

$$\hat{\mu}_{a\to i}^{(t)}(x_i) = \sum_{x_{V(a)\setminus i}} f_a(x_{V(a)}) \prod_{j\in V(a)\setminus i} \mu_{j\to a}^{(t)}(x_j)$$

- variable-i to factor-a message:

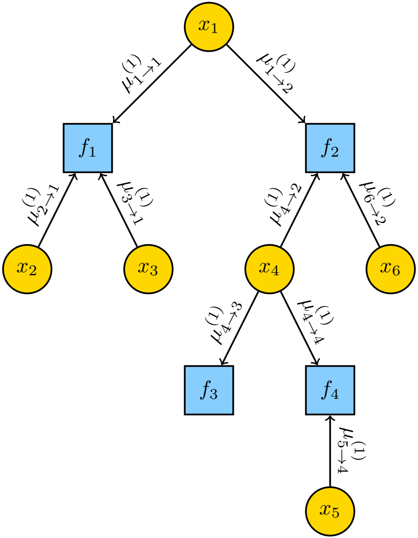
$$\mu_{i\to a}^{(t+1)}(x_i) = \prod_{b\in F(i)\setminus a} \hat{\mu}_{b\to i}^{(t)}(x_i)$$



Marginalization via Message Passing: Example

iteration 1: variable to factor

$$\mu_{i\to a}^{(1)}(x_i) = 1$$



Marginalization via Message Passing: Example

iteration 1: variable to factor

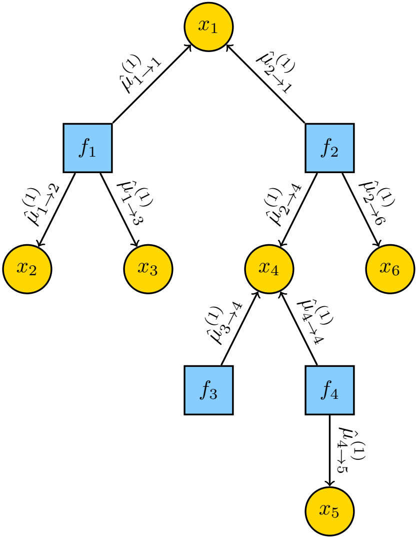
$$\mu_{i\to a}^{(1)}(x_i) = 1$$

iteration 1: factor to variable

$$\hat{\mu}_{4\to 4}^{(1)}(x_4) = \sum_{x_5} f_4(x_4, x_5)\mu_{5\to 4}^{(1)}(x_i)$$

$$= \sum_{x_5} f_4(x_4, x_5)$$

$$\hat{\mu}_{3\to 4}^{(1)}(x_4) = f_3(x_4)$$



Marginalization via Message Passing: Example

iteration 1: factor to variable

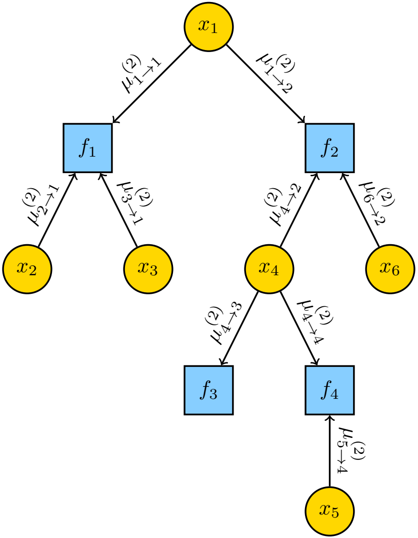
$$\hat{\mu}^{(1)}_{4\to 4}(x_4) = \sum_{x_5} f_4(x_4, x_5)\mu^{(1)}_{5\to 4}(x_i) = \sum_{x_5} f_4(x_4, x_5)$$

$$\hat{\mu}^{(1)}_{3\to 4}(x_4) = f_3(x_4)$$

iteration 2: variable to factor

$$\mu^{(2)}_{4\to 2}(x_4) = \hat{\mu}^{(1)}_{4\to 4}(x_4)\hat{\mu}^{(1)}_{3\to 4}(x_4) = f_3(x_4)\sum_{x_5} f_4(x_4, x_5)$$

$$\mu^{(2)}_{6\to 2}(x_6) = 1$$



Marginalization via Message Passing: Example

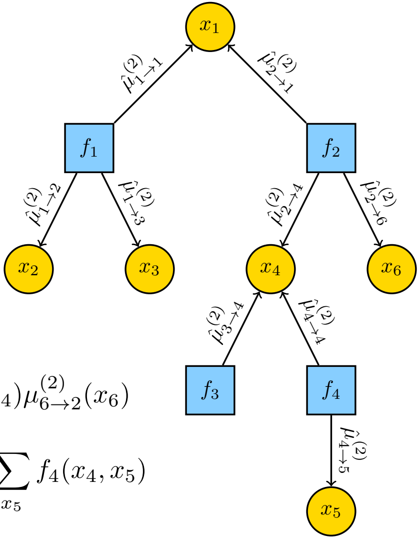
iteration 2: variable to factor

$$\mu^{(2)}_{4\to 2}(x_4) = \hat{\mu}^{(1)}_{4\to 4}(x_4)\hat{\mu}^{(1)}_{3\to 4}(x_4) = f_3(x_4)\sum_{x_5} f_4(x_4,x_5)$$

$$\mu^{(2)}_{6\to 2}(x_6) = 1$$

iteration 2: factor to variable

$$\hat{\mu}^{(1)}_{2\to 1}(x_1) = \sum_{x_4,x_6} f_2(x_1,x_4,x_6)\mu^{(2)}_{4\to 2}(x_4)\mu^{(2)}_{6\to 2}(x_6) = \sum_{x_4,x_6} f_2(x_1,x_4,x_6)f_3(x_4)\sum_{x_5} f_4(x_4,x_5) = f''_2(x_1)$$



Marginalization via Message Passing: Convergence

- For a tree factor graph, consider a FN a and adjacent VN i
  - Let $T(a \to i)$ be the subtree rooted at a given by cutting (i, a)
    - Ex: $T(2 \to 1)$ is subgraph induced by $x_4, x_5, x_6, f_2, f_3, f_4$
  - For all t messages $\hat{\mu}_{a \to i}^{(t)}$ depend only on nodes in $T(a \to i)$
  - Let $h(a \to i)$ be the height of $T(a \to i)$ (i.e., max dist. to leaf)
  - Message $\hat{\mu}_{a \to i}^{(t)}$ converges (i.e., is constant) for $t \geq \left\lceil \frac{h(a \to i)+1}{2} \right\rceil$
  - All messages converge for $t \geq t^* = \left\lceil \frac{\mathrm{diam}(G)}{2} \right\rceil$

Marginalization via Message Passing: Output

- Let $F(a \to i)/V(a \to i)$ be the FNs/VNs in $T(a \to i)$
  - Then, $\hat{\mu}^{(t)}_{a\to i}(x_i)$ converges to the partial sum

$$\hat{\mu}^{(t^*)}_{a\to i}(x_i) = \sum_{x_{V(a\to i)}} \prod_{b\in F(a\to i)} f_b(x_{V(b)})$$

  and the marginalization sum for x_i is given by

$$\begin{aligned} g_i(x_i) &\triangleq \sum_{x_1^n \backslash x_i} f(x_1,\ldots,x_n) = \sum_{x_1^n \backslash x_i} \prod_{b\in F} f_b(x_{V(b)}) \\ &= \sum_{x_1^n \backslash x_i} \prod_{a\in F(i)} \prod_{b\in F(a\to i)} f_b(x_{V(b)}) \\ &= \prod_{a\in F(i)} \sum_{x_{V(a\to i)} \backslash x_i} \prod_{b\in F(a\to i)} f_b(x_{V(b)}) \\ &= \prod_{a\in \partial i} \hat{\mu}^{(t^*)}_{b\to i}(x_i) \end{aligned}$$

Factor Graphs and Probability

- Factor graphs for PMF of random variables $X_1, \ldots, X_n$
  - Any non-negative $f(x_1, \ldots, x_n)$ can be normalized to be a PMF

$$\mathbb{P}\left(X_1^n = x_1^n\right) = \frac{1}{Z} f(x_1, \ldots, x_n),$$

  where $Z = \sum_{x_1^n} f(x_1, \ldots, x_n)$ is the partition function

  - Computing Z has the same complexity as marginalization

- Messages can also be normalized into distributions

$$\overline{\mu}_{i \to a}^{(t)}(x_i) = \frac{\mu_{i \to a}^{(t)}(x_i)}{\sum_{x_i'} \mu_{i \to a}^{(t)}(x_i')}$$

  - Works with message passing: scaled input $\to$ scaled output

Bayesian Networks

- Factor graphs are closely related to Bayesian Networks (BNs)
  - BN is a directed acyclic graph with nodes $V = \{X_1, \ldots, X_n\}$:

$$\mathbb{P}\left(X_1^n = x_1^n\right) = \prod_{i=1}^{n} \mathbb{P}\left(X_i = x_i \,|\, X_{\pi(i)} = x_{\pi(i)}\right),$$

  where $\pi(i)$ is the set of VNs with a directed edge to X_i

- For a tree factor graph, one can build a BN with similar structure
  - If the FG has cycles, then a similar BN may not exist

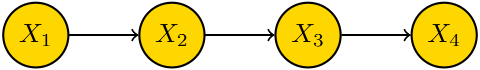
Markov Chain Example

- Factor graph of the Markov chain: $X_1 \to X_2 \to X_3 \to X_4$

$$f(x_1, x_2, x_3, x_4) = f_{12}(x_1, x_2) f_{23}(x_2, x_3) f_{34}(x_3, x_4)$$

- BN of the Markov chain: $X_1 \to X_2 \to X_3 \to X_4$

$$\mathbb{P}\left(X_1^n = x_1^n\right) = \mathbb{P}(X_1 = x_1) \prod_{i=2}^{4} \mathbb{P}\left(X_i = x_i \,|\, X_{i-1} = x_{i-1}\right),$$



Conditional Independence for Factor Graphs

- Let $A, B, S \subset [n]$ be disjoint subsets of VNs in factor graph G
  - If S separates A from B (i.e., there is no path in G from A to B that avoids S), then X_A cond. ind. of X_B given X_S

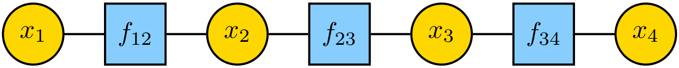
$$P(x_A, x_B | x_S) = P(x_A | x_S) P(x_B | x_S)$$

Conditional Independence for Factor Graphs

- Let $A, B, S \subset [n]$ be disjoint subsets of VNs in factor graph G
  - If S separates A from B (i.e., there is no path in G from A to B that avoids S), then X_A cond. ind. of X_B given X_S

$$P(x_A, x_B | x_S) = P(x_A | x_S) P(x_B | x_S)$$

  - Markov chain example: $A = \{x_1, x_2\}$, $B = \{x_4\}$, $S = \{x_3\}$

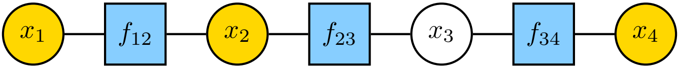


Conditional Independence for Factor Graphs

- Let $A, B, S \subset [n]$ be disjoint subsets of VNs in factor graph G
  - If S separates A from B (i.e., there is no path in G from A to B that avoids S), then X_A cond. ind. of X_B given X_S

$$P(x_A, x_B | x_S) = P(x_A | x_S) P(x_B | x_S)$$

  - Markov chain example: $A = \{x_1, x_2\}$, $B = \{x_4\}$, $S = \{x_3\}$



- Sketch of Proof:
  - Fixing $X_S = x_S$ separates the FG into disjoint components

Conditional Independence for Factor Graphs

- Let $A, B, S \subset [n]$ be disjoint subsets of VNs in factor graph G
  - If S separates A from B (i.e., there is no path in G from A to B that avoids S), then X_A cond. ind. of X_B given X_S

$$P(x_A, x_B | x_S) = P(x_A | x_S) P(x_B | x_S)$$

  - Markov chain example: $A = \{x_1, x_2\}$, $B = \{x_4\}$, $S = \{x_3\}$

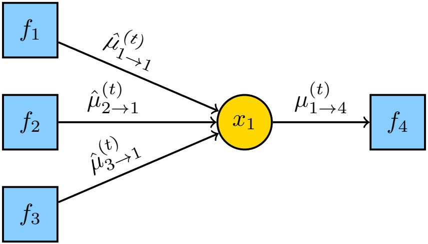
- Sketch of Proof:
  - Fixing $X_S = x_S$ separates the FG into disjoint components
  - Groups of VNs in different components are independent
  - X_A and X_B ind. because A and B in different components

Log Likelihood-Ratio Messages for Binary Variables

- Normalized binary messages given by scalar: $\mu(1) = 1 - \mu(0)$
  - One can also use the likelihood ratio (LR): $\frac{\mu(0)}{\mu(1)}$
  - Or the log likelihood-ratio (LLR): $L = \ln \frac{\mu(0)}{\mu(1)}$
- For inference, LLR messages contain all the information:

$$L_{i \to a}^{(t)} = \ln \frac{\mu_{i \to a}^{(t)}(0)}{\mu_{i \to a}^{(t)}(1)} \qquad \hat{L}_{a \to i}^{(t)} = \ln \frac{\hat{\mu}_{a \to i}^{(t)}(0)}{\hat{\mu}_{a \to i}^{(t)}(1)}$$

VN Update for Binary Variables in LLR Domain



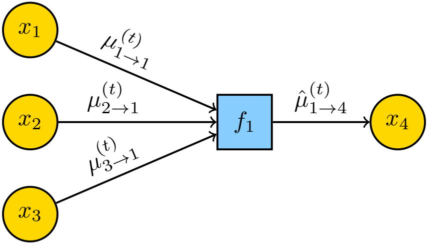
- Recall that the VN message-passing update is:

$$\mu^{(t+1)}_{i\to a}(x_i) = \prod_{b\in F(i)\backslash a} \hat{\mu}^{(t)}_{b\to i}(x_i)$$

- In the LLR domain, this simplifies to

$$L^{(t+1)}_{i\to a} = \ln \frac{\mu^{(t+1)}_{i\to a}(0)}{\mu^{(t+1)}_{i\to a}(1)} = \ln \frac{\prod_{b\in F(i)\backslash a} \hat{\mu}^{(t)}_{b\to i}(0)}{\prod_{b\in F(i)\backslash a} \hat{\mu}^{(t)}_{b\to i}(1)} = \sum_{b\in F(i)\backslash a} \hat{L}^{(t)}_{b\to i}$$

FN Update for Binary Variables in LLR Domain



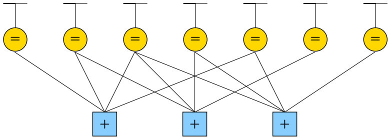
- Recall that the FN message-passing update is:

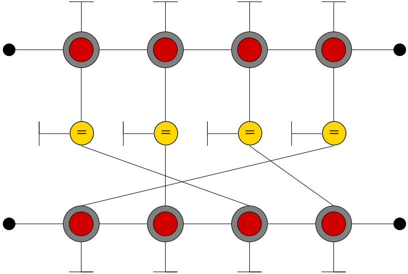
$$\hat{\mu}^{(t)}_{a\to i}(x_i) = \sum_{x_{V(a)}\setminus x_i} f_1(x_{V(a)}) \prod_{j\in V(a)\setminus i} \mu^{(t)}_{j\to a}(x_j)$$

- In the LLR domain, this gives

$$\hat{L}^{(t)}_{a\to i} = \ln \frac{\hat{\mu}^{(t)}_{a\to i}(0)}{\hat{\mu}^{(t)}_{a\to i}(1)} = \ln \frac{\sum_{x_{V(a)}:x_i=0} f_1(x_{V(a)}) \prod_{j\in V(a)\setminus i} \mu^{(t)}_{j\to a}(x_j)}{\sum_{x_{V(a)}:x_i=1} f_1(x_{V(a)}) \prod_{j\in V(a)\setminus i} \mu^{(t)}_{j\to a}(x_j)}$$

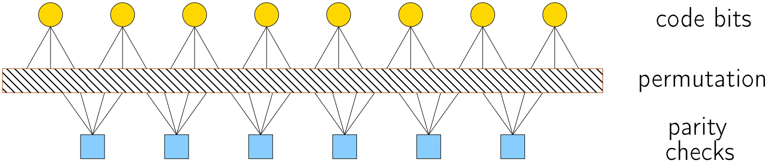
Sparse Graph Codes





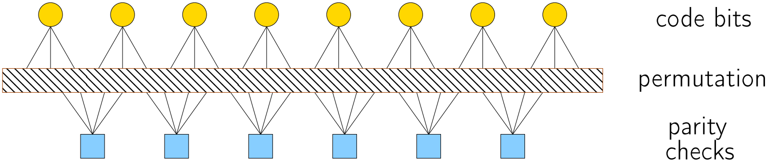
- Codeword constraints defined via sparse factor graph
  - Vertices = constraints
  - Edges = variables, Half-edges = Observations
- Three typical constraints
  - Equality (=): Edges are bits that must have the same value
  - Parity (+): Edges are bits that must sum to zero (mod 2)
  - Trellis: Bit edges must be compatible with state edges

LDPC Codes and Factor Graphs



- Linear codes defined by $xH^T = 0$ for all c.w. $x \in \mathcal{C}$
  - H is an $r \times n$ sparse parity-check matrix for the code
  - Ensembles defined by bit/check degrees and rand. perm.

LDPC Codes and Factor Graphs



- Linear codes defined by $xH^T = 0$ for all c.w. $x \in \mathcal{C}$
  - H is an $r \times n$ sparse parity-check matrix for the code
  - Ensembles defined by bit/check degrees and rand. perm.
- Factor graph: bit/check nodes associated with cols/rows of H
  - Bits are VNs in FG with local factors from channel observations
  - Checks are FNs defined by: $f_{\text{even}}(x_1^d) = I(x_1 \oplus \cdots \oplus x_d = 0)$
  - In this case, indicator function of the code factors

$$\mathbf{1}_{\mathcal{C}}(x_1^n) = \prod_{i=1}^{r} f_{\text{even}}(x_{F(i)})$$

Simplified Summation for Parity Constraints

- Let $f_{\text{even}}(x_1^d) = I(x_1 \oplus \cdots \oplus x_d = 0)$ and consider the expression

$$\phi(z) = \prod_{i=1}^{d} \left(\mu_{i\to}(0) + z\, \mu_{i\to}(1)\right)$$

$$= \sum_{x_1^d \in \{0,1\}^d} \prod_{i=1}^{d} \mu_{i\to}(x_j) z^{w_H(x_1^d)}$$

Simplified Summation for Parity Constraints

- Let $f_{\text{even}}(x_1^d) = I(x_1 \oplus \cdots \oplus x_d = 0)$ and consider the expression

$$\phi(z) = \prod_{i=1}^{d} \left(\mu_{i\to}(0) + z\, \mu_{i\to}(1)\right)$$

$$= \sum_{x_1^d \in \{0,1\}^d} \prod_{i=1}^{d} \mu_{i\to}(x_j) z^{w_H(x_1^d)}$$

- From this, we find the simplified even parity sum:

$$\frac{1}{2}\left[\phi(z) + \phi(-z)\right] = \sum_{x_1^d \in \{0,1\}^d} \prod_{i=1}^{d} \mu_{i\to}(x_j) \frac{1}{2}\left[(1)^{w_H(x_1^d)} + (-1)^{w_H(x_1^d)}\right]$$

$$= \sum_{x_1^d \in \{0,1\}^d} \prod_{i=1}^{d} \mu_{i\to}(x_j)\, I\left(w_H(x_1^d) \text{ is even}\right)$$

$$= \sum_{x_1^d \in \{0,1\}^d} f_{\text{even}}(x_1^d) \prod_{i=1}^{d} \mu_{i\to}(x_j)$$

Simplified Summation for Parity Constraints

- Let $f_{\text{odd}}(x_1^d) = I(x_1 \oplus \cdots \oplus x_d = 1)$ and consider the expression

$$\phi(z) = \prod_{i=1}^{d} \left(\mu_{i\to}(0) + z\, \mu_{i\to}(1) \right)$$

$$= \sum_{x_1^d \in \{0,1\}^d} \prod_{i=1}^{d} \mu_{i\to}(x_j) z^{w_H(x_1^d)}$$

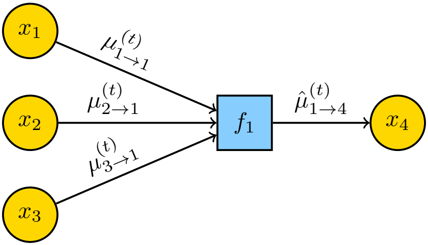
- From this, we find the simplified odd parity sum:

$$\frac{1}{2}\left[\phi(z) - \phi(-z)\right] = \sum_{x_1^d \in \{0,1\}^d} \prod_{i=1}^{d} \mu_{i\to}(x_j) \frac{1}{2}\left[(1)^{w_H(x_1^d)} - (-1)^{w_H(x_1^d)}\right]$$

$$= \sum_{x_1^d \in \{0,1\}^d} \prod_{i=1}^{d} \mu_{i\to}(x_j)\, I\left(w_H(x_1^d) \text{ is odd}\right)$$

$$= \sum_{x_1^d \in \{0,1\}^d} f_{\text{odd}}(x_1^d) \prod_{i=1}^{d} \mu_{i\to}(x_j)$$

FN Update for Even-Parity Constraint in LLR Domain



- In the LLR domain, the FN message-passing update is:

$$\hat{L}^{(t)}_{1\to 4} = \ln \frac{\hat{\mu}^{(t)}_{1\to 4}(0)}{\hat{\mu}^{(t)}_{1\to 4}(1)} = \ln \frac{\sum_{x_1^3} f_{\text{even}}(x_1^3) \prod_{j=1}^{3} \mu^{(t)}_{j\to 1}(x_j)}{\sum_{x_1^3} f_{\text{odd}}(x_1^3) \prod_{j=1}^{3} \mu^{(t)}_{j\to 1}(x_j)}$$

- For degree-d ($d = 4$ above), the simplified sum gives

$$\hat{L}^{(t)}_{1\to d} = \ln \frac{\prod_{j=1}^{d-1} \left(\mu^{(t)}_{j\to 1}(0) + \mu^{(t)}_{j\to 1}(1)\right) + \prod_{j=1}^{d-1} \left(\mu^{(t)}_{j\to 1}(0) - \mu^{(t)}_{j\to 1}(1)\right)}{\prod_{j=1}^{d-1} \left(\mu^{(t)}_{j\to 1}(0) + \mu^{(t)}_{j\to 1}(1)\right) - \prod_{j=1}^{d-1} \left(\mu^{(t)}_{j\to 1}(0) - \mu^{(t)}_{j\to 1}(1)\right)}$$

FN Update for Even-Parity Constraint in LLR Domain

- For degree-d ($d = 4$ above), the simplified sum gives

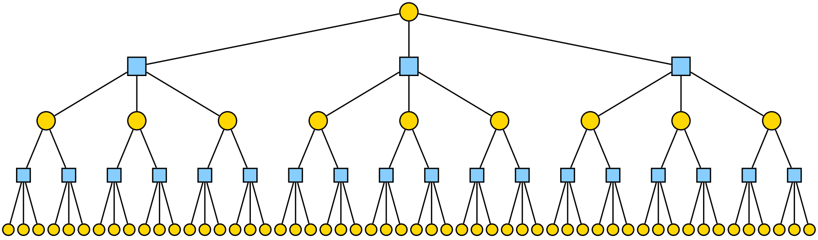
$$\hat{L}^{(t)}_{1\to d} = \ln \frac{\prod_{j=1}^{d-1}\left(\mu^{(t)}_{j\to 1}(0) + \mu^{(t)}_{j\to 1}(1)\right) + \prod_{j=1}^{d-1}\left(\mu^{(t)}_{j\to 1}(0) - \mu^{(t)}_{j\to 1}(1)\right)}{\prod_{j=1}^{d-1}\left(\mu^{(t)}_{j\to 1}(0) + \mu^{(t)}_{j\to 1}(1)\right) - \prod_{j=1}^{d-1}\left(\mu^{(t)}_{j\to 1}(0) - \mu^{(t)}_{j\to 1}(1)\right)}$$

$$= \ln \frac{1 + \prod_{j=1}^{d-1} \frac{\mu^{(t)}_{j\to 1}(0) - \mu^{(t)}_{j\to 1}(1)}{\mu^{(t)}_{j\to 1}(0) + \mu^{(t)}_{j\to 1}(1)}}{1 - \prod_{j=1}^{d-1} \frac{\mu^{(t)}_{j\to 1}(0) - \mu^{(t)}_{j\to 1}(1)}{\mu^{(t)}_{j\to 1}(0) + \mu^{(t)}_{j\to 1}(1)}}$$

$$= \ln \frac{1 + \prod_{j=1}^{d-1} \tanh\left(\frac{1}{2} L^{(t)}_{j\to 1}\right)}{1 - \prod_{j=1}^{d-1} \tanh\left(\frac{1}{2} L^{(t)}_{j\to 1}\right)} \qquad \left(\tanh\left(\tfrac{1}{2}\ln\tfrac{a}{b}\right) = \tfrac{a-b}{a+b}\right)$$

$$= 2\tanh^{-1}\left(\prod_{j=1}^{d-1} \tanh\left(\frac{1}{2} L^{(t)}_{j\to 1}\right)\right) \qquad \left(2\tanh^{-1}(z) = \ln\tfrac{1+z}{1-z}\right)$$

"Leave One Out Rule" for LDPC Decoding



- The "leave one out rule" may seem rather strange at first
  - Similar to the extrinsic decoding rule for turbo codes
- For each bit, it unrolls message passing into a computation graph
  - Inference is exact if computation graph is a tree
- For random LDPC, most VNs have tree-like graphs to depth $\Omega(\ln n)$