

6. Max-product algorithm

- MAP elimination algorithm
- Max-product algorithm on a tree
- Example: LDPC (Low-Density Parity Check) code over BSC (Binary Symmetric Channel)

Inference tasks on graphical models

consider an undirected graphical model (a.k.a. Markov random field)

$$\mu(x) = \frac{1}{Z} \prod_{c \in \mathcal{C}} \psi_c(x_c)$$

where \mathcal{C} is the set of all maximal cliques in G

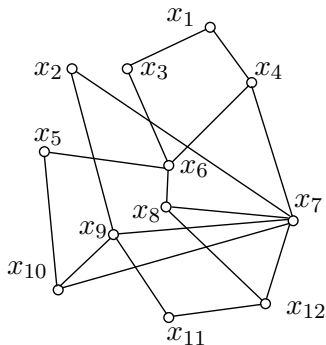
we want to

- calculate marginals: $\mu(x_A) = \sum_{x_{V \setminus A}} \mu(x)$
- calculating conditional distributions

$$\mu(x_A | x_B) = \frac{\mu(x_A, x_B)}{\mu(x_B)}$$

- **calculation maximum a posteriori estimates:** $\arg \max_{\hat{x}} \mu(\hat{x})$
- calculating the partition function Z
- sample from this distribution

MAP estimation = Mode computation



MAP estimation of x given y

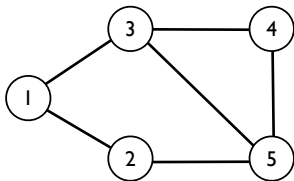
$$x^* \in \arg \max_{x \in \mathcal{X}^V} \mu(x|y)$$

Mode computation

$$\begin{aligned} x^* &\in \arg \max_{x \in \mathcal{X}^V} \mu(x) \\ &= \arg \max_{x \in \mathcal{X}^V} \prod_{c \in \mathcal{C}} \psi_c(x_c) \end{aligned}$$

MAP elimination algorithm for computing the mode

MAP elimination algorithm is exact but can require $O(|\mathcal{X}|^{|V|})$ operations



$$\mu(x) \propto e^{x_1 x_2} e^{-x_1 x_3} e^{x_2 x_5} e^{-x_3 x_4 x_5}$$

- $x_i \in \{+1, -1\}$
- we want to find $x^* \in \arg \max_x \mu(x)$
- brute force combinatorial search:

$$\max_{x \in \{+1, -1\}^5} \mu(x) \propto \max_{x_1, x_2, x_3, x_4, x_5} e^{x_1 x_2} e^{-x_1 x_3} e^{x_2 x_5} e^{-x_3 x_4 x_5}$$

- requires $O(|\mathcal{C}| \cdot |\mathcal{X}|^5)$ operations

- consider an elimination ordering (5, 4, 3, 2, 1)

$$\begin{aligned}
\max_x \mu(x) &\propto \max_{x_1, x_2, x_3, x_4, x_5} e^{x_1 x_2} e^{-x_1 x_3} e^{x_2 x_5} e^{-x_3 x_4 x_5} \\
&= \max_{x_1, x_2, x_3, x_4} e^{x_1 x_2} e^{-x_1 x_3} \underbrace{\max_{x_5 \in \mathcal{X}} e^{(x_2 - x_3 x_4) x_5}}_{\equiv m_5(x_2, x_3, x_4) = e^{|x_2 - x_3 x_4|}, \text{ with } x_5^* = \text{sign}(x_2 - x_3 x_4)} \\
&= \max_{x_1, x_2, x_3, x_4} e^{x_1 x_2} e^{-x_1 x_3} m_5(x_2, x_3, x_4) \\
&= \max_{x_1, x_2, x_3} e^{x_1 x_2} e^{-x_1 x_3} \underbrace{\max_{x_4 \in \mathcal{X}} m_5(x_2, x_3, x_4)}_{\equiv m_4(x_2, x_3) = e^2, \text{ with } x_4^* = -(x_2 x_3)} \\
&= \max_{x_1, x_2, x_3} e^{x_1 x_2} e^{-x_1 x_3} m_4(x_2, x_3) \\
&= \max_{x_1, x_2} e^{x_1 x_2} \underbrace{\max_{x_3 \in \mathcal{X}} e^{-x_1 x_3} m_4(x_2, x_3)}_{\equiv m_3(x_1, x_2) = e^3, \text{ with } x_3^* = -x_1} \\
&= \max_{x_1, x_2} e^{x_1 x_2} m_3(x_1, x_2)
\end{aligned}$$

in the final step, let $(x_1^*, x_2^*) \in \arg \max e^{3+x_1 x_2}$, that is $x_2^* = x_1$ and $x_1^* = +1$ or -1

- from

$x_1^* \in \{\pm 1\}, x_2^* = x_1, x_3^* = -x_1, x_4^* = -x_2x_3, x_5^* = \text{sign}(x_2 - x_3x_4)$,
we can find a MAP assignment by backtracking (5, 4, 3, 2, 1):

$$x^* = (+1, +1, -1, +1, +1) \text{ or } (-1, -1, +1, +1, -1)$$

maximizes $e^{x_1x_2} e^{-x_1x_3} e^{x_2x_5} e^{-x_3x_4x_5}$

- since we are only searching for one maximizing configuration, we are free to break ties arbitrarily
- computational complexity depends on the elimination ordering
- how do we know which ordering is better?

Computational complexity of elimination algorithm

$$\begin{aligned}
 \mu(x_1) &\propto \max_{x_2, x_3, x_4, x_5 \in \mathcal{X}} e^{x_1 x_2} e^{-x_1 x_3} e^{x_2 x_5} e^{-x_3 x_4 x_5} \\
 &= \max_{x_2, x_3, x_4 \in \mathcal{X}} e^{x_1 x_2} e^{-x_1 x_3} \cdot \underbrace{\max_{x_5 \in \mathcal{X}} e^{x_2 x_5} e^{-x_3 x_4 x_5}}_{\equiv m_5(S_5), S_5 = \{x_2, x_3, x_4\}, \Psi_5 = \{e^{x_2 x_5}, e^{-x_3 x_4 x_5}\}} \\
 &= \max_{x_2, x_3 \in \mathcal{X}} e^{x_1 x_2} e^{-x_1 x_3} \cdot \underbrace{\max_{x_4 \in \mathcal{X}} m_5(x_2, x_3, x_4)}_{\equiv m_4(S_4), S_4 = \{x_2, x_3\}, \Psi_4 = \{m_5\}} \\
 &= \max_{x_2 \in \mathcal{X}} e^{x_1 x_2} \cdot \underbrace{\max_{x_3 \in \mathcal{X}} e^{-x_1 x_3} m_4(x_2, x_3)}_{\equiv m_3(S_3), S_3 = \{x_1, x_2\}, \Psi_3 = \{e^{-x_1 x_3}, m_4\}} \\
 &= \underbrace{\max_{x_2 \in \mathcal{X}} e^{x_1 x_2} m_3(x_1, x_2)}_{\equiv m_2(S_2), S_2 = \{x_1\}, \Psi_2 = \{e^{x_1 x_2}, m_3\}} = m_2(x_1)
 \end{aligned}$$

- total complexity: $\sum_i O(|\Psi_i| \cdot |\mathcal{X}|^{1+|S_i|}) = O(|V| \cdot \max_i |\Psi_i| \cdot |\mathcal{X}|^{1+\max_i |S_i|})$
- use induced graph for graph theoretic complexity analysis

MAP elimination algorithm

- **input:** $\{\psi_c\}_{c \in \mathcal{C}}$, alphabet \mathcal{X} , elimination ordering I
 - **output:** $x^* \in \arg \max \mu(x)$
1. initialize active set Ψ to be the set of input compatibility functions
 2. **for** node i in I that is not in A **do**
 - let S_i be the set of nodes, not including i , that share a compatibility function with i
 - let Ψ_i be the set of compatibility functions in Ψ involving x_i
 - compute

$$m_i(x_{S_i}) = \max_{x_i} \prod_{\psi \in \Psi_i} \psi(x_i, x_{S_i})$$

$$x_i^*(x_{S_i}) \in \arg \max_{x_i} \prod_{\psi \in \Psi_i} \psi(x_i, x_{S_i})$$

remove elements of Ψ_i from Ψ

add m_i to Ψ

end

3. produce x^* by traversing I in the reverse order

$$x_{I(j)}^* = x_{I(j)}^*(x_{I(k)}^* : j < k)$$

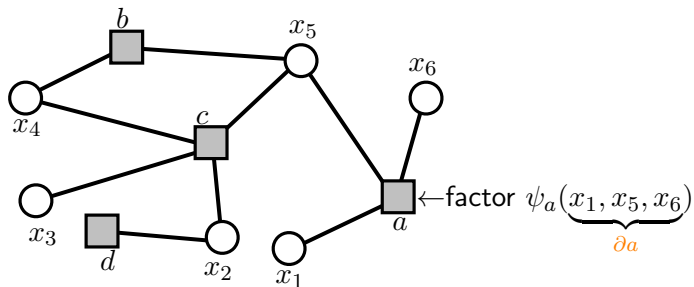
Max-product algorithm

- **max-product algorithm** computes **max-marginal**:

$$\tilde{\mu}_i(x_i) \equiv \max_{x_{V \setminus \{i\}}} \mu(x)$$

- when each max-marginal is uniquely achieved, that is $x_i^* \in \arg \max_{x_i} \tilde{\mu}_i(x_i)$ is unique for all $i \in V$, then $x^* = (x_1^*, \dots, x_n^*)$ is the global MAP configuration
- when there is a tie, we can keep track of 'backpointers' on what maximizing value of the neighbors are to recover a global MAP configuration
- note that in general $\arg \max_{x_i} \tilde{\mu}_i(x_i) \neq \arg \max_{x_i} \mu(x_i)$

Max-product algorithm for factor graphs



$$\mu(x) = \frac{1}{Z} \prod_{a \in F} \psi_a(x_{\partial a})$$

- most generic form of max-product algorithm on a Factor Graph

- ▶ message update:

$$\nu_{i \rightarrow a}^{(t+1)}(x_i) = \prod_{b \in \partial i \setminus \{a\}} \tilde{\nu}_{b \rightarrow i}^{(t)}(x_i)$$

$$\tilde{\nu}_{a \rightarrow i}^{(t)}(x_i) = \max_{x_{\partial a \setminus \{i\}}} \prod_{j \in \partial a \setminus \{i\}} \nu_{j \rightarrow a}^{(t)}(x_j) \psi_a(x_{\partial a})$$

- ▶ backtracking:

$$\delta_{a \rightarrow i}^{(t)}(x_i) = \arg \max_{x_{\partial a \setminus \{i\}}} \prod_{j \in \partial a \setminus \{i\}} \nu_{j \rightarrow a}^{(t)}(x_j) \psi_a(x_{\partial a})$$

- ▶ max-marginal:

$$\tilde{\mu}_i^{(t)}(x_i) = \prod_{a \in \partial i} \tilde{\nu}_{a \rightarrow i}^{(t)}(x_i)$$

- for pairwise MRFs there are two ways to reduce this updates into single update of single messages:

$$\nu_{i \rightarrow j}^{(t+1)}(x_i) = \prod_{k \in \partial i \setminus \{j\}} \left\{ \max_{x_k} \psi(x_k, x_i) \nu_{k \rightarrow i}^{(t)}(x_k) \right\} \quad \tilde{\nu}_{i \rightarrow j}^{(t+1)}(x_j) = \max_{x_i} \psi(x_i, x_j) \prod_{k \in \partial i \setminus \{j\}} \tilde{\nu}_{k \rightarrow i}^{(t)}(x_i)$$

$$\delta_{i \rightarrow j}^{(t+1)}(x_i) = \arg \max_{x_{\partial i \setminus \{j\}}} \prod_{k \in \partial i \setminus \{j\}} \psi(x_k, x_i) \nu_{k \rightarrow i}^{(t)}(x_k) \quad \delta_{i \rightarrow j}^{(t+1)}(x_j) = \arg \max_{x_i} \psi(x_i, x_j) \prod_{k \in \partial i \setminus \{j\}} \tilde{\nu}_{k \rightarrow i}^{(t)}(x_i)$$

- **(parallel) max-product algorithm for pairwise MRFs**

1. initialize $\nu_{i \rightarrow j}^{(0)}(x_i) = 1/|\mathcal{X}|$ for all $(i, j) \in D$
2. for $t \in \{0, 1, \dots, t_{\max}\}$
for all $(i, j) \in D$ compute

$$\nu_{i \rightarrow j}^{(t+1)}(x_i) = \prod_{k \in \partial i \setminus j} \left\{ \max_{x_k} \psi_{ik}(x_i, x_k) \nu_{k \rightarrow i}^{(t)}(x_k) \right\}$$

$$\delta_{i \rightarrow j}^{(t+1)}(x_i) = \arg \max_{x_{\partial i \setminus j}} \prod_{k \in \partial i \setminus j} \left\{ \psi_{ik}(x_i, x_k) \nu_{k \rightarrow i}^{(t)}(x_k) \right\}$$

3. for each $i \in V$ compute max-marginal

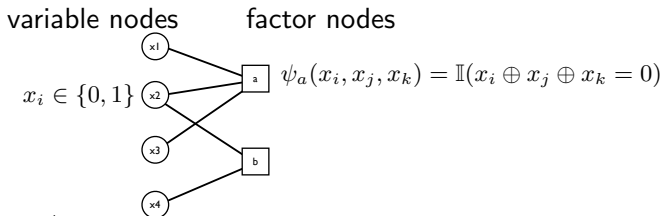
$$\tilde{\mu}_i(x_i) = \prod_{k \in \partial i} \left\{ \max_{x_k} \psi_{ik}(x_i, x_k) \nu_{k \rightarrow i}^{(t_{\max}+1)}(x_k) \right\}$$

$$\delta_i(x_i) = \arg \max_{x_{\partial i}} \prod_{k \in \partial i} \left\{ \psi_{ik}(x_i, x_k) \nu_{k \rightarrow i}^{(t_{\max}+1)}(x_k) \right\}$$

4. backtrack to find a MAP configuration
- related to dynamic programming
 - exact when $t_{\max} \geq \text{diam}(T)$
 - computes max-marginals in $O(n|\mathcal{X}|^2 \cdot \text{diam}(T))$ operations

Example: decoding LDPC codes

- LDPC code is defined by a factor graph model

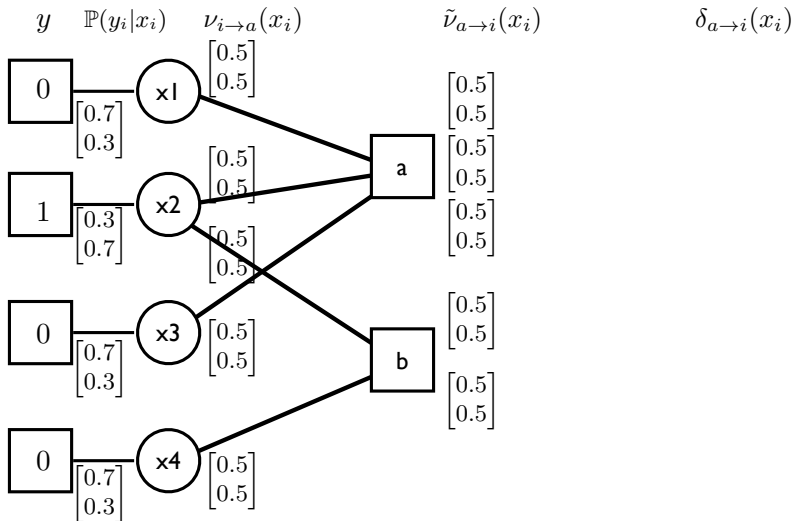


- ▶ block length $n = 4$
 - ▶ number of factors $m = 2$
 - ▶ allowed messages = $\{0000, 0111, 1010, 1101\}$
- decoding using max-product algorithm (for BSC with $\epsilon = 0.3$)

$$\mu_y(x) = \frac{1}{Z} \prod_{i \in V} \mathbb{P}_{Y|X}(y_i|x_i) \prod_{a \in F} \mathbb{I}(\oplus x_{\partial a} = 0)$$

- use (parallel) max-product algorithm to find MAP estimate:

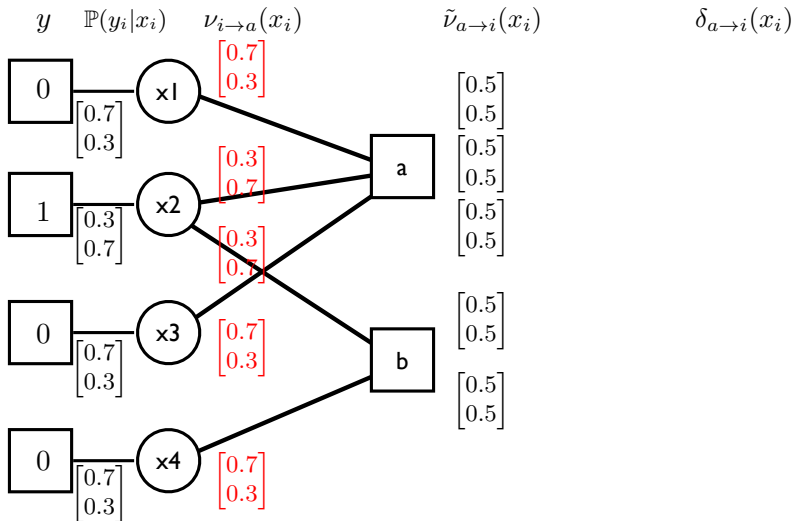
$$\hat{x} = \arg \max \mu(x)$$



$$\nu_{i \rightarrow a}^{(t+1)}(x_i) = \mathbb{P}(y_i|x_i) \prod_{b \in \partial i \setminus \{a\}} \tilde{\nu}_{b \rightarrow i}^{(t)}(x_i)$$

$$\tilde{\nu}_{a \rightarrow i}^{(t+1)}(x_i) = \max_{x_{\partial a \setminus \{i\}}} \prod_{j \in \partial a \setminus \{i\}} \nu_{j \rightarrow a}(x_j) \mathbb{I}(\oplus x_{\partial a} = 0)$$

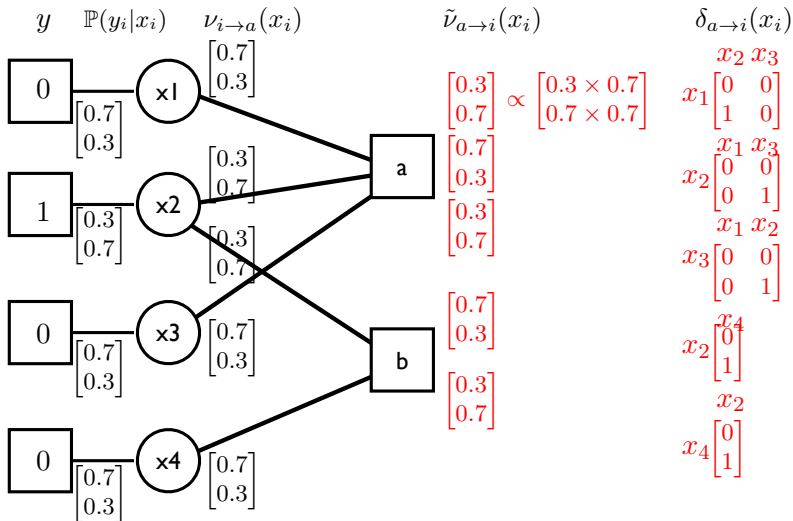
$$\delta_{a \rightarrow i}^{(t)}(x_i) = \arg \max_{x_{\partial a \setminus \{i\}}} \prod_{j \in \partial a \setminus \{i\}} \nu_{j \rightarrow a}^{(t)}(x_j) \mathbb{I}(\oplus x_{\partial a} = 0)$$



$$\nu_{i \rightarrow a}^{(t+1)}(x_i) = \mathbb{P}(y_i|x_i) \prod_{b \in \partial i \setminus \{a\}} \tilde{\nu}_{b \rightarrow i}^{(t)}(x_i)$$

$$\tilde{\nu}_{a \rightarrow i}^{(t+1)}(x_i) = \max_{x_{\partial a \setminus \{i\}}} \prod_{j \in \partial a \setminus \{i\}} \nu_{j \rightarrow a}(x_j) \mathbb{I}(\oplus x_{\partial a} = 0)$$

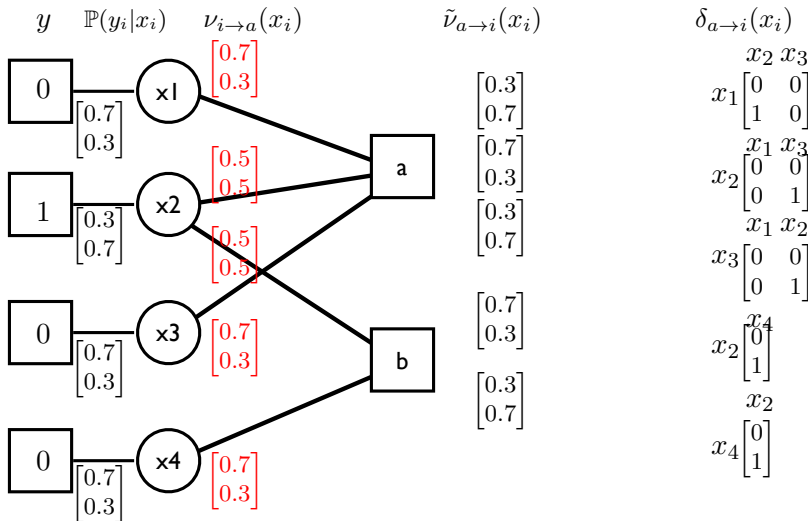
$$\delta_{a \rightarrow i}^{(t)}(x_i) = \arg \max_{x_{\partial a \setminus \{i\}}} \prod_{j \in \partial a \setminus \{i\}} \nu_{j \rightarrow a}^{(t)}(x_j) \mathbb{I}(\oplus x_{\partial a} = 0)$$



$$\nu_{i \rightarrow a}^{(t+1)}(x_i) = \mathbb{P}(y_i|x_i) \prod_{b \in \partial i \setminus \{a\}} \tilde{\nu}_{b \rightarrow i}^{(t)}(x_i)$$

$$\tilde{\nu}_{a \rightarrow i}^{(t+1)}(x_i) = \max_{x_{\partial a \setminus \{i\}}} \prod_{j \in \partial a \setminus \{i\}} \nu_{j \rightarrow a}(x_j) \mathbb{I}(\oplus x_{\partial a} = 0)$$

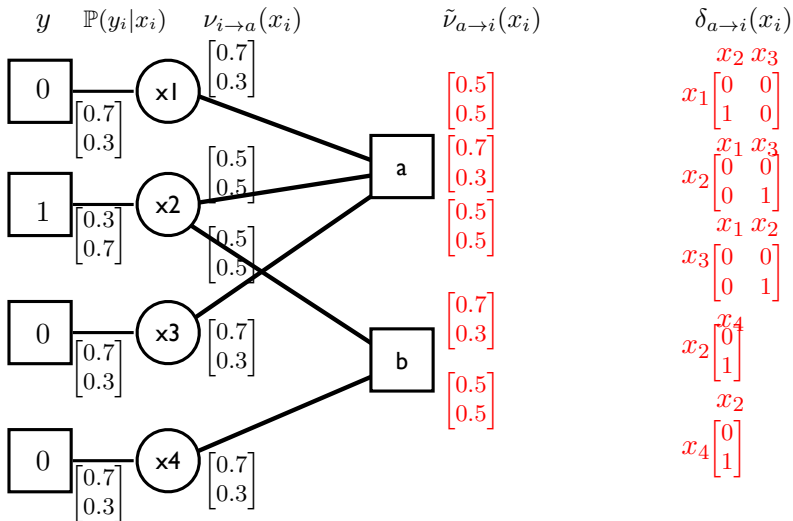
$$\delta_{a \rightarrow i}^{(t)}(x_i) = \arg \max_{x_{\partial a \setminus \{i\}}} \prod_{j \in \partial a \setminus \{i\}} \nu_{j \rightarrow a}^{(t)}(x_j) \mathbb{I}(\oplus x_{\partial a} = 0)$$



$$\nu_{i \rightarrow a}^{(t+1)}(x_i) = \mathbb{P}(y_i|x_i) \prod_{b \in \partial i \setminus \{a\}} \tilde{\nu}_{b \rightarrow i}^{(t)}(x_i)$$

$$\tilde{\nu}_{a \rightarrow i}^{(t+1)}(x_i) = \max_{x_{\partial a \setminus \{i\}}} \prod_{j \in \partial a \setminus \{i\}} \nu_{j \rightarrow a}(x_j) \mathbb{I}(\oplus x_{\partial a} = 0)$$

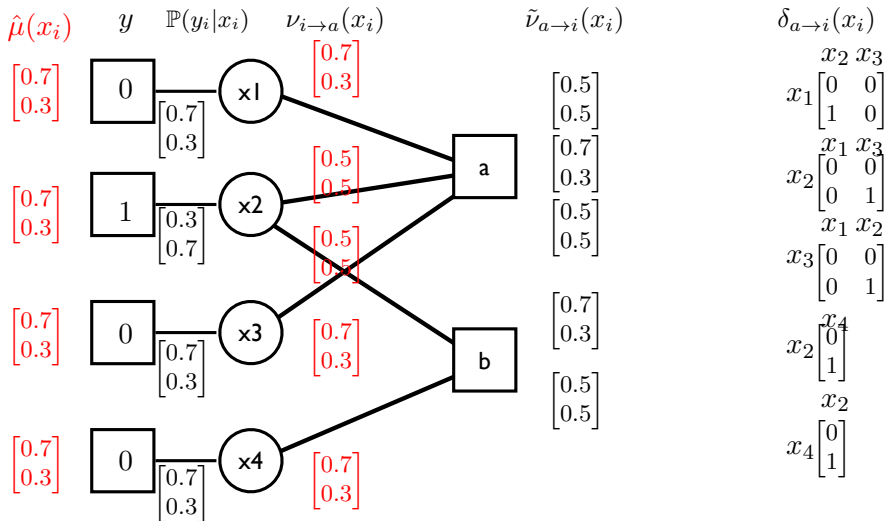
$$\delta_{a \rightarrow i}^{(t)}(x_i) = \arg \max_{x_{\partial a \setminus \{i\}}} \prod_{j \in \partial a \setminus \{i\}} \nu_{j \rightarrow a}^{(t)}(x_j) \mathbb{I}(\oplus x_{\partial a} = 0)$$



$$\nu_{i \rightarrow a}^{(t+1)}(x_i) = \mathbb{P}(y_i|x_i) \prod_{b \in \partial i \setminus \{a\}} \tilde{\nu}_{b \rightarrow i}^{(t)}(x_i)$$

$$\tilde{\nu}_{a \rightarrow i}^{(t+1)}(x_i) = \max_{x_{\partial a \setminus \{i\}}} \prod_{j \in \partial a \setminus \{i\}} \nu_{j \rightarrow a}(x_j) \mathbb{I}(\oplus x_{\partial a} = 0)$$

$$\delta_{a \rightarrow i}^{(t)}(x_i) = \arg \max_{x_{\partial a \setminus \{i\}}} \prod_{j \in \partial a \setminus \{i\}} \nu_{j \rightarrow a}^{(t)}(x_j) \mathbb{I}(\oplus x_{\partial a} = 0)$$



$$\nu_{i \rightarrow a}^{(t+1)}(x_i) = \mathbb{P}(y_i|x_i) \prod_{b \in \partial i \setminus \{a\}} \tilde{\nu}_{b \rightarrow i}^{(t)}(x_i)$$

$$\tilde{\nu}_{a \rightarrow i}^{(t+1)}(x_i) = \max_{x_{\partial a \setminus \{i\}}} \prod_{j \in \partial a \setminus \{i\}} \nu_{j \rightarrow a}(x_j) \mathbb{I}(\oplus x_{\partial a} = 0)$$

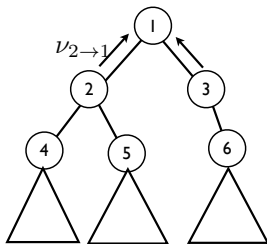
$$\delta_{a \rightarrow i}^{(t)}(x_i) = \arg \max_{x_{\partial a \setminus \{i\}}} \prod_{j \in \partial a \setminus \{i\}} \nu_{j \rightarrow a}^{(t)}(x_j) \mathbb{I}(\oplus x_{\partial a} = 0)$$

Numerical stability and min-sum algorithm

- multiplying many compatibility functions can lead to numerical issues (e.g. when the values are close to zero)
- one remedy is to work in log domain
- **min-sum algorithm**

$$\nu_{i \rightarrow j}(x_i) = \sum_{k \in \partial i \setminus j} \min_{x_k} \left\{ -\log \psi_{ki}(x_k, x_i) + \nu_{k \rightarrow i}(x_k) \right\}$$

$$\tilde{\mu}_i(x_i) = \exp \left\{ \sum_{k \in \partial i} \min_{x_k} \left\{ -\log \psi_{ki}(x_k, x_i) + \nu_{k \rightarrow i}(x_k) \right\} \right\}$$

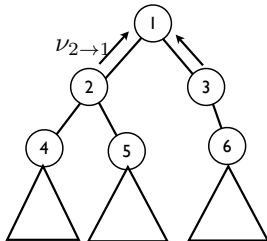


- define graphical model on sub-trees

$T_{2 \rightarrow 1} = (V_{2 \rightarrow 1}, E_{2 \rightarrow 1}) \equiv$ Subtree rooted at 2 and excluding 1

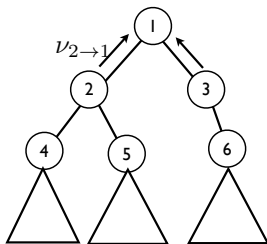
$$\mu_{2 \rightarrow 1}(x_{V_{2 \rightarrow 1}}) \equiv \frac{1}{Z(T_{2 \rightarrow 1})} \prod_{(u,v) \in E_{2 \rightarrow 1}} \psi(x_u, x_v)$$

$$\nu_{2 \rightarrow 1}(x_2) \equiv \max_{x_{V_{2 \rightarrow 1} \setminus \{2\}}} \mu_{2 \rightarrow 1}(x_{V_{2 \rightarrow 1}})$$



- the messages from neighbors are sufficient to compute the max-marginal $\tilde{\mu}(x_1)$

$$\begin{aligned}
 \tilde{\mu}(x_1) &\propto \max_{x_{V \setminus \{1\}}} \prod_{(u,v) \in E} \psi(x_u, x_v) \\
 &= \max_{x_{V_{2 \rightarrow 1}}, x_{V_{3 \rightarrow 1}}} \psi(x_1, x_2) \psi(x_1, x_3) \left\{ \prod_{(u,v) \in E_{2 \rightarrow 1}} \psi(x_u, x_v) \right\} \left\{ \prod_{(u,v) \in E_{3 \rightarrow 1}} \psi(x_u, x_v) \right\} \\
 &= \max_{x_{V_{2 \rightarrow 1}}, x_{V_{3 \rightarrow 1}}} \prod_{k \in \partial 1} \left\{ \psi(x_1, x_k) \left\{ \prod_{(u,v) \in E_{k \rightarrow 1}} \psi(x_u, x_v) \right\} \right\} \\
 &= \prod_{k \in \partial 1} \max_{x_{V_{k \rightarrow 1}}} \left\{ \psi(x_1, x_k) \left\{ \prod_{(u,v) \in E_{k \rightarrow 1}} \psi(x_u, x_v) \right\} \right\} \\
 &= \prod_{k \in \partial 1} \max_{x_k} \left\{ \psi(x_1, x_k) \underbrace{\left\{ \max_{x_{V_{k \rightarrow 1} \setminus \{k\}}} \prod_{(u,v) \in E_{k \rightarrow 1}} \psi(x_u, x_v) \right\}}_{\nu_{k \rightarrow 1}(x_k)} \right\}
 \end{aligned}$$



- recursion on sub-trees to compute the messages ν

$$\begin{aligned}
 \mu_{2 \rightarrow 1}(x_{V_{2 \rightarrow 1}}) &= \frac{1}{Z(T_{2 \rightarrow 1})} \prod_{(u,v) \in E_{2 \rightarrow 1}} \psi(x_u, x_v) \\
 &\propto \psi(x_2, x_4) \psi(x_2, x_5) \left\{ \prod_{(u,v) \in E_{4 \rightarrow 2}} \psi(x_u, x_v) \right\} \left\{ \prod_{(u,v) \in E_{5 \rightarrow 2}} \psi(x_u, x_v) \right\} \\
 &\propto \psi(x_2, x_4) \psi(x_2, x_5) \mu_{4 \rightarrow 2}(x_{V_{4 \rightarrow 2}}) \mu_{5 \rightarrow 2}(x_{V_{5 \rightarrow 2}})
 \end{aligned}$$

$$\begin{aligned}
\nu_{2 \rightarrow 1}(x_2) &= \max_{x_{V_{2 \rightarrow 1} \setminus 2}} \mu_{2 \rightarrow 1}(x_{V_{2 \rightarrow 1}}) \\
&\propto \max_{x_{V_{2 \rightarrow 1} \setminus 2}} \psi(x_2, x_4) \psi(x_2, x_5) \mu_{4 \rightarrow 2}(x_{V_{4 \rightarrow 2}}) \mu_{5 \rightarrow 2}(x_{V_{5 \rightarrow 2}}) \\
&\propto \left\{ \max_{x_{V_{4 \rightarrow 2}}} \psi(x_2, x_4) \mu_{4 \rightarrow 2}(x_{V_{4 \rightarrow 2}}) \right\} \left\{ \max_{x_{V_{5 \rightarrow 2}}} \psi(x_2, x_5) \mu_{5 \rightarrow 2}(x_{V_{5 \rightarrow 2}}) \right\} \\
&= \left\{ \max_{x_4} \psi(x_2, x_4) \max_{x_{V_{4 \rightarrow 2} \setminus \{4\}}} \mu_{4 \rightarrow 2}(x_{V_{4 \rightarrow 2}}) \right\} \left\{ \max_{x_5} \psi(x_2, x_5) \max_{x_{V_{5 \rightarrow 2} \setminus \{5\}}} \mu_{5 \rightarrow 2}(x_{V_{5 \rightarrow 2}}) \right\} \\
&\propto \left\{ \max_{x_4} \psi(x_2, x_4) \nu_{4 \rightarrow 2}(x_4) \right\} \left\{ \max_{x_5} \psi(x_2, x_5) \nu_{5 \rightarrow 2}(x_5) \right\}
\end{aligned}$$

- update messages via recursion with uniform initialization $\nu_{i \rightarrow j}(x_i) = \frac{1}{|\mathcal{X}|}$ for all leaves i

$$\nu_{2 \rightarrow 1}(x_2) = \prod_{k \in \partial 2 \setminus \{1\}} \left\{ \max_{x_k} \psi_{2k}(x_2, x_k) \nu_{k \rightarrow 2}(x_k) \right\}$$

- to obtain a global configuration, we need to store the $\arg \max$ for each message for backtracking

$$\delta_{2 \rightarrow 1}(x_2) = \arg \max_{x_{\partial 2 \setminus \{1\}}} \prod_{k \in \partial 2 \setminus \{1\}} \left\{ \psi_{2k}(x_2, x_k) \nu_{k \rightarrow 2}(x_k) \right\}$$