

Sum-Product Algorithm

Seungjin Choi

Department of Computer Science
POSTECH, Korea
seungjin@postech.ac.kr

1

Sum-Product Algorithm

- A method of probabilistic inference that only works on trees.
 - Can compute the marginal probabilities of all nodes with the same time complexity as computing the marginal probability of just one node.
 - A node-to-node message passing protocol that reuses messages between computing marginal probabilities.
- Works for both directed and undirected graphical models.
- For the case of directed graphical models, the algorithm can be used only if their **moralized graph** is a **tree** or **polytree**.

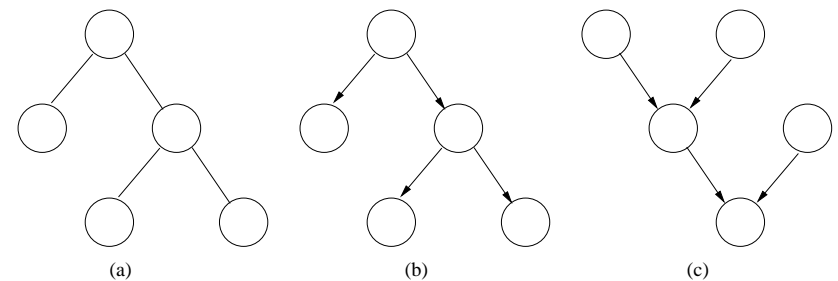
3

Outline

- Sum-product algorithm
- Max-product algorithm
- Factor graphs

2

Trees



(a) An undirected tree: One and only one path between any pair of nodes.

(b) A directed tree: A directed graph whose moralized graph is an undirected tree (at most one parent).

(c) A polytree: Its moralized graph has cycles.

A directed tree and the corresponding undirected tree make exactly the same conditional independence assumptions, so we cover them together.

4

Parameterization and Conditioning on $\mathcal{G}(\mathcal{V}, \mathcal{E})$

Parameterization

- **Undirected trees** $p(x) = \frac{1}{Z} \left[\prod_{i \in \mathcal{V}} \psi(x_i) \prod_{(i,j) \in \mathcal{E}} \psi(x_i, x_j) \right]$
- **Directed trees** $p(x) = p(x_r) \prod_{(i,j) \in \mathcal{E}} p(x_j | x_i)$

Given $p(x_r)$ and $p(x_j | x_i)$ of a directed graph, the potentials of the corresponding undirected graph can be parameterized as:

$$\psi(x_i) = \begin{cases} p(x_r) & i = r \\ 1 & i \neq r \end{cases}, \quad \psi(x_i, x_j) = p(x_j | x_i). \quad \Rightarrow \quad Z = 1$$

Conditioning

$$\psi^E(x_i) \equiv \begin{cases} \psi(x_i) \delta(x_i, \bar{x}_i) = \delta(x_i, \bar{x}_i) & i \in E \\ \psi(x_i) = 1 & i \notin E \end{cases}$$

$$p(x | \bar{x}_E) = \frac{1}{Z^E} \left[\prod_{i \in \mathcal{V}} \psi^E(x_i) \prod_{(i,j) \in \mathcal{E}} \psi(x_i, x_j) \right]$$

5

Elimination: A Review

1. Choose an **elimination ordering** I in which the query node f is the final node.
2. Place all potentials on an **active list**.
3. Eliminate a node i by removing all potentials referencing the node from the active list, taking the product, summing over x_i , and placing the resulting intermediate factor back on the active list.

$$\begin{aligned} p(x_1, \bar{x}_6) &= \frac{1}{Z} \sum_{x_2} \sum_{x_3} \sum_{x_4} \sum_{x_5} \sum_{x_6} \psi(x_1, x_2) \psi(x_1, x_3) \psi(x_2, x_4) \psi(x_3, x_5) \psi(x_2, x_5, x_6) \delta(x_6, \bar{x}_6) \\ &= \frac{1}{Z} \sum_{x_2} \psi(x_1, x_2) \sum_{x_3} \psi(x_1, x_3) \sum_{x_4} \psi(x_2, x_4) \sum_{x_5} \psi(x_3, x_5) \sum_{x_6} \psi(x_2, x_5, x_6) \delta(x_6, \bar{x}_6) \\ &= \frac{1}{Z} \sum_{x_2} \psi(x_1, x_2) \sum_{x_3} \psi(x_1, x_3) \sum_{x_4} \psi(x_2, x_4) \sum_{x_5} \psi(x_3, x_5) m_6(x_2, x_5) \\ &= \frac{1}{Z} \sum_{x_2} \psi(x_1, x_2) \sum_{x_3} \psi(x_1, x_3) m_5(x_2, x_3) \sum_{x_4} \psi(x_2, x_4) \\ &= \frac{1}{Z} \sum_{x_2} \psi(x_1, x_2) m_4(x_2) \sum_{x_3} \psi(x_1, x_3) m_5(x_2, x_3) \\ &= \frac{1}{Z} \sum_{x_2} \psi(x_1, x_2) m_4(x_2) m_3(x_1, x_2) \\ &= \frac{1}{Z} m_2(x_1) \end{aligned}$$

7

Note that ...

The parameterization of **unconditional distributions** on **trees** is given by

$$p(x) = \frac{1}{Z} \left[\prod_{i \in \mathcal{V}} \psi(x_i) \prod_{(i,j) \in \mathcal{E}} \psi(x_i, x_j) \right].$$

The parameterization of **conditional distributions** on **trees** is given by

$$p(x | \bar{x}_E) = \frac{1}{Z^E} \left[\prod_{i \in \mathcal{V}} \psi^E(x_i) \prod_{(i,j) \in \mathcal{E}} \psi(x_i, x_j) \right].$$

The parameterization of unconditional distributions and conditional distributions on trees, is formally identical, involving a product of potential functions associated with each node and each edge in the graph.

6

Special Features on Trees

Elimination Orderings **depth-first traversal** of the tree.

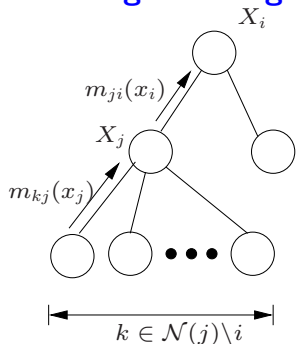
- Treat f as a root and view the tree as a directed tree by directing all edges of the tree to point away from f .
- Any elimination ordering in which a node is eliminated only after all of its children in the directed version of the tree are eliminated.
- All subtrees with no evidence nodes can be ignored.

Elimination Step the **intermediate factor** that is created when a node is eliminated

- Which nodes appear in the potential created after summing over j ?
 - nothing in the subtree below j (already eliminated)
 - nothing outside the subtree, due to the assumption that the graph is tree (For a node k in the subtree and a node l outside of the subtree, there can be no potential $\psi(x_k, x_l)$ in the probability model.)
 - only i from $\psi(x_i, x_j)$

8

Message-Passing



Messages: $m_{ji}(x_i) = \sum_{x_j} \left[\psi^E(x_j) \psi(x_i, x_j) \prod_{k \in \mathcal{N}(j) \setminus i} m_{kj}(x_j) \right]$.

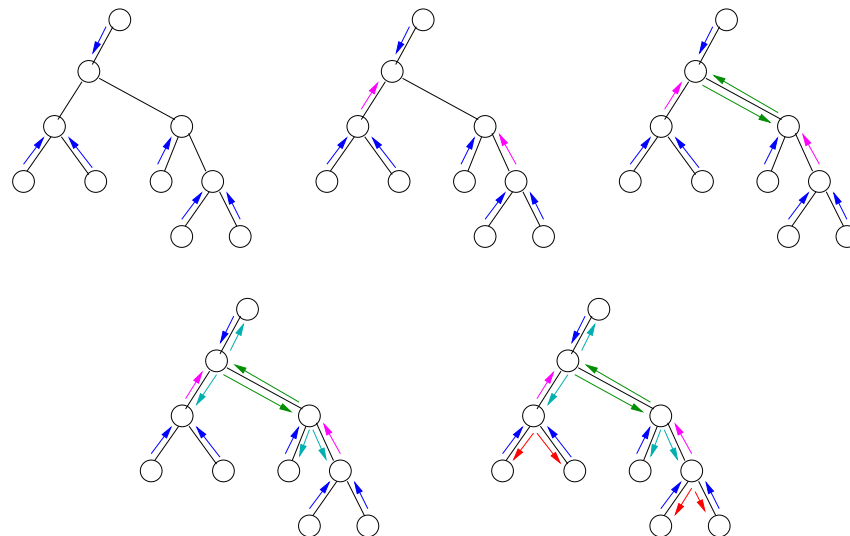
Final node x_f : $p(x_f | \bar{x}_E) \propto \psi^E(x_f) \prod_{e \in \mathcal{N}(f)} m_{ef}(x_f)$.

Computational complexity: $\mathcal{O}(m^2)$ for each message if x_j has m states and $\mathcal{O}(2n)$ for final node where n is the number of edges.

Message-passing protocol: A node can send a message to a neighboring node when it has received messages from all of its other neighbors.

9

Message-Passing Protocol



10

Algorithm Outline: Sum-Product (Belief Propagation)

1. Choose a root node (arbitrarily or as first query node).
2. If j is an evidence node, $\psi^E(x_j) = \delta(x_j, \bar{x}_j)$, else $\psi^E(x_j) = 1$.
3. Pass messages from leaves up to root and then back down using

$$m_{ji}(x_i) = \sum_{x_j} \left[\psi^E(x_j) \psi(x_i, x_j) \prod_{k \in \mathcal{N}(j) \setminus i} m_{kj}(x_j) \right].$$

4. Given messages, compute the marginal of x_f using

$$p(x_f | \bar{x}_E) \propto \psi^E(x_f) \prod_{k \in \mathcal{N}(f)} m_{kf}(x_f)$$

11

Key Insights

- Messages can be **reused**.
- Can obtain all marginals by simply doubling the amount of work required to compute a single marginal.
- The effect of computing over all possible elimination orderings (a huge number) by computing all possible messages (a small number).

12

Maximum a Posteriori (MAP) Configuration

The problem of **MAP** is to find the maximal probability that can be achieved by some set of random variables (x_F), given a set of observation (x_E), where (E, F) is a partition of the indices, $x = (x_1, \dots, x_n)$.

1. Find the maximal probability:

$$\begin{aligned} \max_{x_F} p(x_F | \bar{x}_E) &= \max_{x_F} p(x_F, \bar{x}_E) \\ &= \max_x p(x) \delta(x_E, \bar{x}_E) \\ &= \max_x p^E(x). \end{aligned}$$

2. Find a configuration that achieves the maximal probability:

$$x^* \in \arg \max_x p^E(x).$$

13

MAP: An Example (Cont'd)

We define **messages**, m_{ji}^{\max} , similarly as for the marginalization case, except that all **sum** operators are replaced by **max** operators:

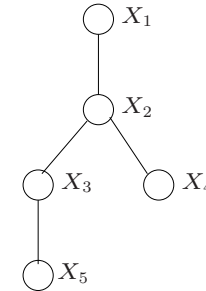
$$\begin{aligned} m_{53}^{\max}(x_3) &= \max_{x_5} \psi^E(x_5) \psi(x_3, x_5), \\ m_{42}^{\max}(x_2) &= \max_{x_4} \psi^E(x_4) \psi(x_2, x_4), \\ m_{32}^{\max}(x_2) &= \max_{x_3} \psi^E(x_3) \psi(x_2, x_3) m_{53}^{\max}(x_3). \end{aligned}$$

Then the maximum probability is achieved by

$$\max_x p(x) = \max_{x_1} \psi^E(x_1) m_{21}^{\max}(x_1).$$

15

MAP: An Example



$$\begin{aligned} \max_x p(x) &= \max_{x_1} \max_{x_2} \max_{x_3} \max_{x_4} \max_{x_5} p(x_1) p(x_2 | x_1) p(x_3 | x_2) p(x_4 | x_2) p(x_5 | x_3) \\ &= \max_{x_1} p(x_1) \max_{x_2} p(x_2 | x_1) \max_{x_3} p(x_3 | x_2) \max_{x_4} p(x_4 | x_2) \max_{x_5} p(x_5 | x_3). \end{aligned}$$

14

Maximum Configuration

Given two nodes i and j such that i is the parent of j , we need to record the maximizing values in a table $\delta_{ji}(x_i)$, which is computed as:

$$\delta_{ji}(x_i) \in \arg \max_{x_j} \psi^E(x_j) \psi(x_i, x_j) \prod_{k \in \mathcal{N}(j) \setminus i} m_{kj}^{\max}(x_j),$$

which just becomes a table in the dimension of x_i in the discrete case.

The root node then becomes

$$x_f^* \in \arg \max_{x_f} \psi^E(x_f) \prod_{k \in \mathcal{N}(f) \setminus i} m_{kf}^{\max}(x_f).$$

Once x_f is computed, for neighbor k of the root node, $x_k^* = \delta_{kf}(x_f^*)$. Starting this propagation at the root node and continuing until all nodes have been hit, will yield the maximum configuration.

16

Algorithm Outline: Max-Product

Max-Product Version of the Algorithm: inward pass (leaves \rightarrow root)

$$m_{ji}^{\max} = \max_{x_j} \left[\psi^E(x_j) \psi(x_i, x_j) \prod_{k \in (N)(j) \setminus i} m_{kj}^{\max}(x_j) \right],$$
$$\max_x p^E(x) = \max_{x_i} \left[\psi^E(x_i) \prod_{j \in (N)(i)} m_{ji}^{\max}(x_i) \right].$$

MAP configurations inward pass and outward pass

$$x_f^* \in \arg \max_{x_f} \left[\psi^E(x_f) \prod_{e \in \mathcal{N}(f)} m_{ef}^{\max}(x_f) \right],$$
$$\delta_{ji}(x_i) \in \arg \max_{x_j} \left[\psi^E(x_j) \psi(x_i, x_j) \prod_{k \in \mathcal{N}(j) \setminus i} m_{kj}^{\max}(x_j) \right].$$

It is often safer to work with the log of probabilities:

$$\log m_{ji}^{\max} = \max_{x_j} \left[\log \psi^E(x_j) + \log \psi(x_i, x_j) + \sum_{k \in (N)(j) \setminus i} \log m_{kj}^{\max}(x_j) \right]$$