

COMPSCI 688: Probabilistic Graphical Models

Lecture 9: Message Passing

Dan Sheldon

Manning College of Information and Computer Sciences
University of Massachusetts Amherst

Partially based on materials by Benjamin M. Marlin (marlin@cs.umass.edu) and Justin Domke (domke@cs.umass.edu)

1 / 22

Message Passing in Chains

2 / 22

Message Passing Derivation

Let's go back to our chain example. Suppose we want to compute $p(x_4)$? Which variables should we eliminate, and in what order?

3 / 22

4 / 22

What if we want to compute $p(x_3)$? Which variables should we eliminate, and in what order?

5 / 22

Message Passing Derivation

When doing “leaf-first” variable elimination to compute any marginal $p(x_i)$, there are only 6 different intermediate factors

$$m_{1 \rightarrow 2}, m_{2 \rightarrow 3}, m_{3 \rightarrow 4}, \quad m_{4 \rightarrow 3}, m_{3 \rightarrow 2}, m_{2 \rightarrow 1}$$

Let's call $m_{j \rightarrow i}$ the “message” from j to i .

We can compute Z by “collecting” messages at any node:

$$Z = \sum_{x_i} \phi_i(x_i) \prod_{j \in \text{nb}(i)} m_{j \rightarrow i}(x_i)$$

The general formula for a marginal is similar, but we omit the final summation and normalize:

$$p(x_i) = \frac{1}{Z} \phi_i(x_i) \prod_{j \in \text{nb}(i)} m_{j \rightarrow i}(x_i)$$

6 / 22

Message Passing Derivation

The messages satisfy recurrences, e.g.

$$m_{2 \rightarrow 3}(x_3) = \sum_{x_2} m_{1 \rightarrow 2}(x_2) \phi_2(x_2) \phi_{23}(x_2, x_3)$$

The message $m_{i-1 \rightarrow i}(x_i)$ sums out all variables from the product of all factors “to the left” of x_i

The message $m_{i+1 \rightarrow i}(x_i)$ has a similar recurrence, and sums out variables/factors “to the right”.

Using the recurrences, we can compute *all messages*, and therefore *all marginals* in two passes through the chain, one in each direction.

7 / 22

Message Passing in a Chain

- ▶ Initialize $m_{0 \rightarrow 1}(x_1) = 1$, $m_{n+1 \rightarrow n}(x_n) = 1$.
- ▶ For $i = 2$ to n
 - ▶ Let $m_{i-1 \rightarrow i}(x_i) = \sum_{x_{i-1}} m_{i-2 \rightarrow i-1}(x_{i-1}) \phi_{i-1}(x_{i-1}) \phi_{i-1,i}(x_{i-1}, x_i)$
- ▶ For $i = n - 1$ down to 1
 - ▶ Let $m_{i+1 \rightarrow i}(x_i) = \sum_{x_{i+1}} m_{i+2 \rightarrow i+1}(x_{i+1}) \phi_{i+1}(x_{i+1}) \phi_{i,i+1}(x_i, x_{i+1})$
- ▶ Compute each unnormalized marginal as $\hat{p}(x_i) = m_{i-1 \rightarrow i}(x_i) \phi_i(x_i) m_{i+1 \rightarrow i}(x_i)$
- ▶ Compute $Z = \sum_{x_i} \hat{p}(x_i)$ for any i , and normalize each marginal: $p(x_i) = \frac{1}{Z} \hat{p}(x_i)$

8 / 22

Pairwise Marginals

- ▶ Correct formula for a pairwise marginal $p(x_i, x_{i+1})$?

$$p(x_i, x_{i+1}) = \frac{1}{Z} m_{i-1 \rightarrow i}(x_i) \phi_i(x_i) \phi_{i,i+1}(x_i, x_{i+1}) \phi_{i+1}(x_{i+1}) m_{i+2 \rightarrow i+1}(x_{i+1})$$

Discussion: Message Passing vs. Variable Elimination

- ▶ Variable elimination can compute marginals and Z **exponentially faster** than direct summation for nice enough graphs (e.g. chains, trees)
- ▶ Naively, to compute all single-node marginals you would have to run variable elimination n times, once per node (but this would repeat work)
- ▶ Message passing can compute all the marginals for the same cost as running variable elimination twice, so is a **factor of $\approx n/2$ faster** than naive variable elimination
- ▶ (Message passing is nice, but you could say variable elimination did the heavy lifting.)

Message Passing in Trees

Message Passing in Trees

A more general version of message passing works for any *tree-structured MRF*, that is, an MRF of the following form where $G = (V, E)$ is a tree:

$$p(\mathbf{x}) = \prod_{i \in V} \phi_i(x_i) \prod_{(i,j) \in E} \phi_{ij}(x_i, x_j).$$

Message passing can be derived from variable elimination. Take x_i as the root and eliminate variables from leaf to root. We get

$$Z = \sum_{x_i} \phi_i(x_i) \prod_{j \in \text{nb}(i)} m_{j \rightarrow i}(x_i)$$

$$p(x_i) = \frac{1}{Z} \phi_i(x_i) \prod_{j \in \text{nb}(i)} m_{j \rightarrow i}(x_i)$$

The “message” $m_{j \rightarrow i}(x_i)$ is the result of summing out all factors and variables in the subtree T_j rooted at x_j .

By similar reasoning, the pairwise marginal for $(i, j) \in E$ is

$$p(x_i, x_j) = \frac{1}{Z} \phi_i(x_i) \phi_{ij}(x_i, x_j) \phi_j(x_j) \prod_{k \in \text{nb}(i) \setminus j} m_{k \rightarrow i}(x_i) \prod_{\ell \in \text{nb}(j) \setminus i} m_{\ell \rightarrow j}(x_j)$$

Recurrence for Messages

The messages satisfy the following recurrence

$$m_{j \rightarrow i}(x_i) = \sum_{x_j} \phi_j(x_j) \phi_{ij}(x_i, x_j) \prod_{k \in \text{nb}(j) \setminus i} m_{k \rightarrow j}(x_j)$$

This can be understood by expanding the summation over T_j to group factors for subtrees rooted at each child of x_j , that is, for each node $k \in \text{nb}(j) \setminus i$.

Message-Passing

Importantly, the message from j to i doesn't depend on which particular node is the root. There are only $2(n-1)$ total messages and we can compute them all in two passes through the tree.

Say that j is **ready to send to i** if j has received messages from all $k \in \text{nb}(j) \setminus i$.

Message passing: while any node j is ready to send to i , compute $m_{j \rightarrow i}$ using recurrence from previous slide.

This algorithm is described asynchronously (“ready-to-send”), but in practice: pass messages from leaves to root of tree and back.

Message-Passing Summary

$$m_{j \rightarrow i}(x_i) = \sum_{x_j} \phi_j(x_j) \phi_{ij}(x_i, x_j) \prod_{k \in \text{nb}(j) \setminus i} m_{k \rightarrow j}(x_j)$$

$$Z = \sum_{x_i} \phi_i(x_i) \prod_{j \in \text{nb}(i)} m_{j \rightarrow i}(x_i)$$

$$p(x_i) = \frac{1}{Z} \phi_i(x_i) \prod_{j \in \text{nb}(i)} m_{j \rightarrow i}(x_i)$$

$$p(x_i, x_j) = \frac{1}{Z} \phi_i(x_i) \phi_{ij}(x_i, x_j) \phi_j(x_j) \prod_{k \in \text{nb}(i) \setminus j} m_{k \rightarrow i}(x_i) \prod_{\ell \in \text{nb}(j) \setminus i} m_{\ell \rightarrow j}(x_j) \quad (i, j) \in E$$

17 / 22

Discussion and Extensions

18 / 22

Discussion

- ▶ Message-passing computes *all single and pairwise marginals* at roughly 2x cost of variable elimination
- ▶ It is restricted to pairwise MRFs and trees, but can be extended in some ways
- ▶ For exactly answering *one query* in any MRF, variable elimination is faster than message passing
- ▶ For exactly answering a set of marginal queries, variable elimination usually takes at most a factor of $O(n)$ more time

19 / 22

Sketches of Extensions

- ▶ What if the MRF has factors on more than two variables? (keyword: *factor graphs*)

20 / 22

- ▶ What if the MRF is not tree-structured, i.e., G has cycles?
- ▶ **Answer 1:** group nodes (keyword: *clique trees* or *junction trees*)

- ▶ What if the MRF is not tree-structured, i.e., G has cycles?
- ▶ **Answer 2:** use message-passing as a fixed-point iteration (keyword: *loopy belief propagation*)