AIRR - Handout on Belief Propagation

Nishanth Kumar and Ethan Yang

Last Revised: 02/15/2025

1 Belief Propagation Overview

Let's try to work through an example to understand the core idea of belief propagation as used in the sum-product algorithm. Specifically, we are interested in computing the marginal distribution of a particular root variable.

Consider this simple example (slightly modified from lecture notes) with 3 binary variables (each can take on value 'T' and 'F').

Factor $f_1(A, C)$

A	C	$f_1(A, C)$
T	T	0.05
T	F	0.45
F	T	0.45
F	F	0.05

Factor $f_2(B, C)$

B	C	$f_2(B, C)$
T	T	0.0
T	F	0.5
F	T	0.0
F	F	0.5

Factor $f_3(A)$

A	$f_3(A)$
T	1
F	1

Factor $f_4(B)$

B	$f_4(B)$
T	1
F	1

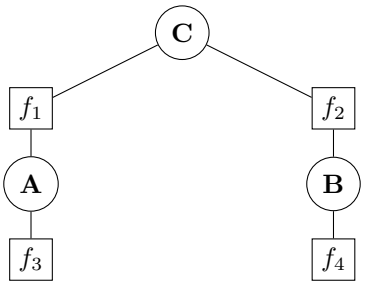

Figure 1: Example Factor Graph FG1

1.1 Computing $P(C)$ the slow way

We know:

$$\begin{aligned} P(C) &= \sum_{V \setminus C} P(V) \\ &= \sum_a \sum_b P(a, b, c) \\ &\propto \sum_a \sum_b f_1(A, C) f_2(B, C) f_3(A) f_4(B) \end{aligned}$$

Note here that a, b, and c are all possible values that variables A, B, and C can take on in turn.

Let's start by computing $P(C)$ via 'brute force'. Specifically, let's build the joint probability distribution explicitly and then compute the marginal on variable C.

$$P(A, B, C) \propto$$

A	B	C	$P(A, B, C) \times Z$
T	T	T	$0.05 \times 0.0 \times 1 \times 1 = 0$
T	T	F	$0.45 \times 0.5 \times 1 \times 1 = 0.225$
T	F	T	$0.05 \times 0.0 \times 1 \times 1 = 0$
T	F	F	$0.45 \times 0.5 \times 1 \times 1 = 0.225$
F	T	T	$0.45 \times 0.0 \times 1 \times 1 = 0.0$
F	T	F	$0.05 \times 0.5 \times 1 \times 1 = 0.025$
F	F	T	$0.45 \times 0.0 \times 1 \times 1 = 0$
F	F	F	$0.05 \times 0.5 \times 1 \times 1 = 0.025$

What we did here is just index into specific values from the different factors. Specifically, for the first entry (A=T, B=T, C=T), we need to multiply the values of all factors that have an 'opinion' (i.e., entry) for each of these variables set to the value T — $f_1(T, T) \times f_2(T, T) \times f_3(T) \times f_4(T)$. Similarly, for the entry (A=F, B=T, C=T) which is the 5th row in this above table, we'd do $f_1(F, T) \times f_2(T, T) \times f_3(F) \times f_4(T)$.

Importantly, these values in the table above compute $P(A, B, C) \times Z$. It isn't a proper probability distribution: we need to normalize by the constant Z. We can compute this by simply summing up all the entries in the final column (0.5) and dividing through all entries by this:

A	B	C	$P(A, B, C) \times Z$
T	T	T	0
T	T	F	0.45
T	F	T	0
T	F	F	0.45
F	T	T	0
F	T	F	0.05
F	F	T	0
F	F	F	0.05

Now, we can compute the distribution on C by simply marginalizing out the other variables.

$$P(C) =$$

C	$P(C)$
T	$0 + 0 + 0 + 0 = 0.0$
F	$0.45 + 0.55 + 0.05 + 0.05 = 1.0$

This strategy will always work: we can always compute a marginal by first computing a joint and then marginalizing out all the variables (in this case A and B) except the one we care about (C). Unfortunately, this approach is inefficient. The size of the joint table (for the case where all variables are binary) is $2^{\#\text{variables}}$. Thus, this procedure will be exponential in general. If we have a lot of variables, actually computing this will be hopeless.

1.2 The more efficient way: Belief Propagation

As in many other instances you'll see in this class, we will gain efficiency here by leveraging *structure* in the problem. In particular, we'll leverage the inherent tree structure of the graph. Recall that we assume that we're running BP on a *tree*: this means that the branches do not connect to one another. In the above example, the 'f_1' branch on the left that has the variable A is entirely unconnected to the 'f_2' branch on the right. Intuitively, this means that we should be able to compute the marginal on C by computing some distribution on the left branch and then combining it with some distribution on the right branch. Since each branch necessarily involves a smaller subset of variables and factors than the entire problem, this procedure is more efficient than trying to compute the entire joint using all variables and all factors.

Let's start from the formula we used above for $P(C)$:

$$\begin{aligned} P(c) &= \sum_{V \setminus C} P(\bar{v}) \\ &= \sum_a \sum_b P(a, b, c) \\ &\propto \sum_a \sum_b f_1(a, c) f_2(b, c) f_3(a) f_4(b) \end{aligned}$$

Note here that $f_1(a, b)f_2(b, c)f_3(a)f_4(b)$ denotes 'table multiplication' (i.e., computing the joint table by multiplying all the individual tables for f_1, f_2, f_3, f_4.

Now, looking at this, we see that f_1 and f_3 are the only factors dependent on variable A and f_2 and f_4 are the only ones dependent on B. Indeed, this is evident from the connectivity of the tree.

Given this, let's just commute the terms above:

$$\sum_a \sum_b f_1(a, c) f_3(a) f_2(b, c) f_4(b)$$

Now, since the first two terms don't depend on B and the second two don't depend on A, we can rewrite this as:

$$\left(\sum_a f_1(a, c) f_3(a) \right) \cdot \left(\sum_b f_2(b, c) f_4(b) \right)$$

What this tells us is that we can *independently* compute a joint table on f_1 and f_3 separately from one on f_2 and f_4, then combine these together! At its core this is because *variable independence*.

In the lecture notes, this is denoted by: $\prod_{\phi \in \mathrm{N}(V_i)} \sum_{\bar{v}_{\mathrm{subtree}(\phi)}} F_\phi(\bar{v})$ (The F_ϕ is exactly each term inside the two summations).

Let's see what those those two tables look like:

Factor $f_{(1,3)} = f_1(A, C) \times f_3(A)$

a	c	$f_1(a, c) \times f_3(a)$
T	T	$0.05 \times 1 = 0.05$
T	F	$0.45 \times 1 = 0.45$
F	T	$0.45 \times 1 = 0.45$
F	F	$0.05 \times 1 = 0.05$

Factor $f_{(2,4)} = f_2(b, c) \times f_4(b)$

b	c	$f_2(b, c) \times f_4(b)$
T	T	$0.0 \times 1 = 0.0$
T	F	$0.5 \times 1 = 0.5$
F	T	$0.0 \times 1 = 0.0$
F	F	$0.5 \times 1 = 0.5$

Now that we have the two tables for the neighbors of C, we can compute the marginal on C by simply producting the marginals in either of these two tables! (Notice that these marginals are exactly the messages $\mu_{f_1 \to C}$ and $\mu_{f_2 \to C}$. We'll explain this more in the next section.)

Message $\mu_{f_1 \to C}$ **from marginalizing** $f_1 \times f_3$

C	$\mu_{f_1 \to C}$
T	$0.05 + 0.45 = 0.5$
F	$0.45 + 0.05 = 0.5$

Message $\mu_{f_2 \to C}$ from marginalizing $f_2 \times f_4$

C	$\mu_{f_2 \to C}$
T	$0.0 + 0.0 = 0.0$
F	$0.5 + 0.5 = 1.0$

Combining these together with $P(C) \propto \mu_{f_1 \to C} \cdot \mu_{f_2 \to C}$

C	Belief(C)
T	$(0.5 \times 0.0) = 0.0$
F	$(0.5 \times 1.0) = 0.5$

Now we've obtained the 'belief' on C. We can get the final $P(C)$ by computing the normalizing constant $Z = 0.5$ and dividing through by this!

C	$P(C)$
T	$(\frac{0.0}{0.5}) = 0.0$
F	$(\frac{0.5}{0.5}) = 1.0$

Notice that the largest table we had to write down in this procedure was only of size 4. In general, sum-product will run in time $O(N \cdot 2^{\text{degree of largest factor}})$, which is much better than our brute force $O(2^N)$.

2 Message Passing in Sum-Product

In this above example, there were only two subtrees, each of depth 2 (i.e., they had two factors in them). What if there were more subtrees, each with larger depth? Hopefully you can see from these examples that the same principle applies: we can break down the computation into separate computations for each of the independent subtrees.

As an example of this, consider another example factor graph FG2 from lecture 4 slides below.

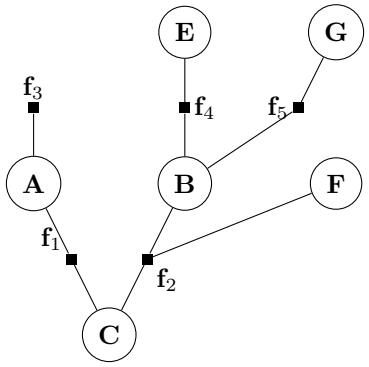

Figure 2: Example factor graph FG2

We can tackle this general case by recognizing that our 'reasoning about subtrees' strategy developed above is *recursive*. Say we want to find the marginal $P(C)$. We can repeatedly factor the joint, shown below.

$$
\begin{aligned}
P(c) &\propto \sum_{a,b,e,f,g} f_1(a,c) f_2(c,b,f) f_3(a) f_4(b,e) f_5(b,g) \\
&= \left(\sum_a f_1(a,c) f_3(a) \right) \left(\sum_{b,e,f,g} f_2(c,b,f) f_4(b,e) f_5(b,g) \right) \\
&= \left(\sum_a f_1(a,c) f_3(a) \right) \left(\sum_{b,f} f_2(c,b,f) \sum_{e,g} f_4(b,e) f_5(b,g) \right) \\
&= \left(\sum_a f_1(a,c) f_3(a) \right) \left(\sum_{b,f} f_2(c,b,f) \left(\sum_e f_4(b,e) \right) \left(\sum_g f_5(b,g) \right) \right) \\
&= \underbrace{\left(\sum_a f_1(a,c) f_3(a) \right)}_{f_1 \text{ subtree, } \mu_{f_1 \to C}} \underbrace{\left(\sum_{b,f} f_2(c,b,f) \underbrace{\left(\sum_e f_4(b,e) \right)}_{f_4 \text{ subtree, } \mu_{f_4 \to B}} \underbrace{\left(\sum_g f_5(b,g) \right)}_{f_5 \text{ subtree, } \mu_{f_5 \to B}} \right)}_{B \text{ subtree, } \mu_{B \to f_2}}}_{f_2 \text{ subtree, } \mu_{f_2 \to C}}
\end{aligned}
$$

Notice, that the last expression is factored in terms of subtrees! Each nested layer recurses deeper into the tree. The factorization depicts exactly how sum-product computes the contribution of each subtree to $P(C)$ independently.

This core idea — of breaking down our probability computation by independent subtrees — gives rise to the message passing procedure within the sum-product algorithm. Repeatedly applying the recursive procedure brings us to a leaf of the tree, and we can then pass information back up the tree in the form of messages.

Let's now look at each of the two message types, and how to compute the marginal of any variable from the messages.

Factor-to-Variable Messages Factor-to-variable messages describe how a factor node ϕ sends information to one of its neighboring variable nodes v. These messages account for the factor's local function and all incoming messages from ϕ's children.

Specifically, a factor ϕ computes a message to a variable v as:

$$
\mu_{\phi \to v}(v) = \sum_{\bar{w}_{N(\phi) \setminus v}} \phi(v, \bar{w}) \prod_{w \in N(\phi) \setminus v} \mu_{w \to \phi}(w)
$$

where:

- $N(\phi)$ is the set of variables connected to factor ϕ,
- $\sum_{\bar{w}_{N(\phi) \setminus v}}$ marginalizes out all variables connected to ϕ except v,
- $\phi(v, \bar{w})$ is the factor table encoding dependencies between the variables,
- $\prod_{w \in N(\phi) \setminus v} \mu_{w \to \phi}(w)$ accounts for incoming messages from all children.

If ϕ is a leaf (i.e., it connects to only one variable), there are no incoming messages to aggregate, so:

$$
\mu_{\phi \to v}(v) = \phi(v)
$$

which simplifies to just the factor table itself.

Variable-to-Factor Messages A variable V computes a message to a factor ϕ as:

$$
\mu_{V \to \phi}(v) = \prod_{\psi \in \mathcal{N}(V) \setminus \phi} \mu_{\psi \to V}(v)
$$

where:

- $\mathcal{N}(V)$ is the set of factors connected to variable V,
- $\mu_{\psi \to V}(v)$ represents messages from neighboring factors

This equation tells us that a variable node collects all incoming messages from its other factors and passes the product of these messages to the target factor. Intuitively, this message expresses how much the subtree rooted at V supports the value v, excluding any influence from ϕ.

If V is a leaf (i.e., it has only one neighboring factor), there are no other incoming messages to aggregate, so:

$$\mu_{V \to \phi}(v) = 1$$

which represents a uniform belief before receiving any influence from factors.

Computing the marginal Finally, the marginal of any variable is proportional to the product of all incoming factor messages.

$$P(V = v) = \prod_{\phi \in N(V)} \mu_{\phi \to V}(v).$$

This formula should look familiar to you: it's exactly what we used in our FG1 example to compute the marginal distribution on C!

$$P(C) = \left(\sum_A f_1(A, C) f_3(A) \right) \cdot \left(\sum_B f_2(B, C) f_4(B) \right)$$

We can decompose this into messages:

$$\mu_{f_1 \to C}(C) = \sum_A f_1(A, C) \mu_{A \to f_1}(A)$$

$$\mu_{f_2 \to C}(C) = \sum_B f_2(B, C) \mu_{B \to f_2}(B)$$

where:

- $\mu_{A \to f_1}(A) = f_3(A)$
- $\mu_{B \to f_2}(B) = f_4(B)$

FG1 Messages Example Walkthrough Let's now work through all the messages required to compute $P(C)$ from our FG1 example from Section 1.2. This is doing the same computation as before, but we are labeling where the messages are.

We start by computing **messages from factors to variables** at the **leaf nodes**.

Factor f_3 to Variable A

$$\mu_{f_3 \to A}(A) = f_3(A)$$

A	$\mu_{f_3 \to A}(A)$
T	1
F	1

Factor f_4 to Variable B

$$\mu_{f_4 \to B}(B) = f_4(B)$$

B	$\mu_{f_4 \to B}(B)$
T	1
F	1

Factor f_1 to Variable C

$$\mu_{f_1 \to C}(C) = \sum_A f_1(A, C) \mu_{f_3 \to A}(A)$$

C	$\mu_{f_1 \to C}(C)$
T	0.5
F	0.5

Factor f_2 to Variable C

$$\mu_{f_2 \to C}(C) = \sum_B f_2(B, C) \mu_{f_4 \to B}(B)$$

C	$\mu_{f_2 \to C}(C)$
T	0.0
F	1.0

Given these, we can compute the marginal distribution on C exactly as we did when walking through belief propagation!

$$b(C) = \mu_{f_1 \to C}(C) \cdot \mu_{f_2 \to C}(C)$$

C	$b(C)$
T	0.0
F	0.5

$$P(C) = \frac{b(C)}{\sum b(C)}$$

C	$P(C)$
T	0.0
F	1.0

Now, let's think about the messages from C back 'down' the tree. These aren't necessary to compute the marginal distribution on C, but we can use them to compute marginals for A and B! This is another reason why sum-product is powerful. If implemented carefully, we can get the marginal for *every* variable in the same time complexity, $O(N \cdot 2^{\text{degree of largest factor}})$.

Variable C to Factor f_1

$$\mu_{C \to f_1}(C) = \mu_{f_2 \to C}(C)$$

C	$\mu_{C \to f_1}(C)$
T	0.0
F	1.0

Variable C to Factor f_2

$$\mu_{C \to f_2}(C) = \mu_{f_1 \to C}(C)$$

C	$\mu_{C \to f_2}(C)$
T	0.5
F	0.5

Factor f_1 to Variable A

$$\mu_{f_1 \to A}(A) = \sum_C f_1(A, C) \mu_{C \to f_1}(C)$$

A	$\mu_{f_1 \to A}(A)$
T	0.45
F	0.05

Factor f_2 to Variable B

$$\mu_{f_2 \to B}(B) = \sum_C f_2(B, C) \mu_{C \to f_2}(C)$$

B	$\mu_{f_2 \to B}(B)$
T	0.25
F	0.25

These are all possible messages in the tree FG1! We can now compute marginals on A and B if we'd like.

Marginal on A

$$b(A) = \mu_{f_3 \to A}(A) \cdot \mu_{f_1 \to A}(A)$$

A	$b(A)$
T	0.45
F	0.05

$$P(A) = \frac{b(A)}{\sum b(A)}$$

A	$P(A)$
T	0.9
F	0.1

Marginal on B

$$b(B) = \mu_{f_4 \to B}(B) \cdot \mu_{f_2 \to B}(B)$$

$$P(B) = \frac{b(B)}{\sum b(B)}$$

B	$P(B)$
T	0.5
F	0.5