Review

Fancy formulas from algebra

$$\underbrace{\sum_{x_1, x_2, \ldots, x_N}}_{\text{all values of each}} f(x_1, x_2, \ldots, x_N) = \underbrace{\sum_{x_1} \sum_{x_2} \ldots \sum_{x_N}}_{\text{any order you like}} f(x_1, x_2, \ldots, x_N)$$

(essentially a definition)

$$\left(\sum a_i\right)\left(\sum b_j\right) = \sum \sum a_i b_j$$

(exchanging products and sums)

Review

More warmup

$$\sum_{x_2} \sum_{x_1} \underbrace{\psi(x_2, x_3)}_{\substack{\text{No dependency on } x_1 \\ \text{hence we can move} \\ \text{factor outside sum} \\ \text{over } x_1.}} \psi(x_1, x_2) = \sum_{x_2} \psi(x_2, x_3) \underbrace{\sum_{x_1} \psi(x_1, x_2)}_{\substack{\text{Vector of size over the} \\ \text{components of } x_2}}$$

This is **not** the same as

$$\left\{\sum_{x_2} \psi(x_2, x_3)\right\}\left\{\sum_{x_1} \psi(x_1, x_2)\right\}$$

because x_2 is shared!

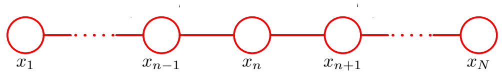
(Recall the distibutive law: $ba + ca = a(b+c)$)

This rule enables us to move sums “inwards” (or equivalently factors “outward”) to break big sums over big products into smaller pieces.

This works as long as what is being shuffled do not have variables in common (e.g., sum over x_1 and potential over x_2 and x_3).

Review

Marginals on a chain



$$p(\mathrm{x}) = \psi_{1,2}(x_1, x_2)\psi_{2,3}(x_2, x_3) \quad \ldots \quad \psi_{N-2,N-1}(x_{N-2}, x_{N-1})\psi_{N-1,N}(x_{N-1}, x_N)$$

$$p(x_n) = \sum_{x_1} \sum_{x_2} \ldots \sum_{x_{n-1}} \sum_{x_{n+1}} \ldots \sum_{x_{N-1}} \sum_{x_N} f_L(x_1, x_1, \ldots, x_n) f_R(x_n, x_{n+1}, \ldots, x_N)$$

$$= \left(\sum_{x_1} \sum_{x_2} \ldots \sum_{x_{n-1}} f_L(x_1, x_1, \ldots, x_n)\right)\left(\sum_{x_{n+1}} \ldots \sum_{x_{N-1}} \sum_{x_N} f_R(x_n, x_{n+1}, \ldots, x_N)\right)$$

$$= \left(\sum_{x_1} \sum_{x_2} \ldots \sum_{x_{n-1}} \prod_{i=1}^{n-1} \psi_{i,i+1}(x_i, x_{i+1})\right)\left(\sum_{x_{n+1}} \ldots \sum_{x_{N-1}} \sum_{x_N} \prod_{i=n}^{N-1} \psi_{i,i+1}(x_i, x_{i+1})\right)$$

$$= \left(\sum_{x_{n-1}} \sum_{x_{n-2}} \ldots \sum_{x_1} \prod_{i=1}^{n-1} \psi_{n-i,n-i+1}(x_{n-i}, x_{n-i+1})\right)\left(\sum_{x_{n+1}} \ldots \sum_{x_{N-1}} \sum_{x_N} \prod_{i=n}^{N-1} \psi_{i,i+1}(x_i, x_{i+1})\right)$$

Review

$$p(x_n) = \left(\sum_{x_{n-1}} \sum_{x_{n-2}} \ldots \sum_{x_1} \prod_{i=1}^{n-1} \psi_{n-i,n-i+1}(x_{n-i}, x_{n-i+1})\right)\left(\sum_{x_{n+1}} \ldots \sum_{x_{N-1}} \sum_{x_N} \prod_{i=n}^{N-1} \psi_{i,i+1}(x_i, x_{i+1})\right)$$

$$\sum_{x_{n-1}} \sum_{x_{n-2}} \ldots \sum_{x_1} \prod_{i=1}^{n-1} \psi_{n-i,n-i+1}(x_{n-i}, x_{n-i+1}) = \sum_{x_{n-1}} \sum_{x_{n-2}} \ldots \sum_{x_2} \prod_{i=1}^{n-2} \psi_{n-i,n-i+1}(x_{n-i}, x_{n-i+1})\left\{\sum_{x_1} \psi_{1,2}(x_1, x_2)\right\}$$

$$= \sum_{x_{n-1}} \sum_{x_{n-2}} \ldots \sum_{x_3} \prod_{i=1}^{n-3} \psi_{n-i,n-i+1}(x_{n-i-1}, x_{n-i})\left\{\sum_{x_2} \psi_{2,3}(x_2, x_3)\left\{\sum_{x_1} \psi_{1,2}(x_1, x_2)\right\}\right\}$$

$\ldots$

$$= \left\{\sum_{x_{n-1}} \psi_{n-1,n}(x_{n-1}, x_n) \ldots \left\{\sum_{x_3} \psi_{3,4}(x_3, x_4)\left\{\sum_{x_2} \psi_{2,3}(x_2, x_3)\left\{\sum_{x_1} \psi_{1,2}(x_1, x_2)\right\}\right\}\right\}\right\}$$

Review

$$p(x_n)=\left(\sum_{x_{n-1}}\sum_{x_{n-2}}\cdots\sum_{x_1}\prod_{i=1}^{n-1}\psi_{n-i,\,n-i+1}(x_{n-i},x_{n-i+1})\right)\left(\sum_{x_{n+1}}\cdots\sum_{x_{N-1}}\sum_{x_N}\prod_{i=n}^{N-1}\psi_{i,\,i+1}(x_i,x_{i+1})\right)$$

where

$$\sum_{x_{n-1}}\sum_{x_{n-2}}\cdots\sum_{x_1}\prod_{i=1}^{n-1}\psi_{n-i,n-i+1}(x_{n-i},x_{n-i+1})=\left\{\sum_{x_{n-1}}\psi_{n-1,n}(x_{n-1},x_n)\cdots\left\{\sum_{x_3}\psi_{3,4}(x_3,x_4)\left\{\sum_{x_2}\psi_{2,3}(x_2,x_3)\left\{\sum_{x_1}\psi_{1,2}(x_1,x_2)\right\}\right\}\right\}\cdots\right\}$$

and

$$\sum_{x_{n+1}}\cdots\sum_{x_{N-1}}\sum_{x_N}\prod_{i=n}^{N-1}\psi_{i,i+1}(x_i,x_{i+1})=\left\{\sum_{x_{n+1}}\psi_{n,n+1}(x_n,x_{n+1})\cdots\left\{\sum_{x_{N-1}}\psi_{N-2,N-1}(x_{N-2},x_{N-1})\left\{\sum_{x_N}\psi_{N-1,N}(x_{N-1},x_N)\right\}\right\}\cdots\right\}$$

(Deriving the right factor is similar to doing the left one which we did in detail.)

Review

Computational Complexity

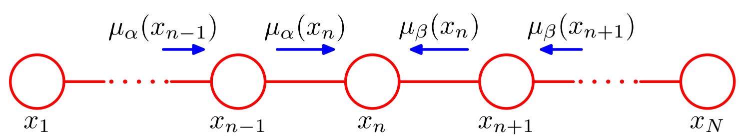
Suppose each variable has K values
What is the cost of evaluating the first factor?

$$\sum_{x_{n-1}}\sum_{x_{n-2}}\cdots\sum_{x_1}\prod_{i=1}^{n-1}\psi_{n-i,n-i+1}(x_{n-i},x_{n-i+1})=\left\{\sum_{x_{n-1}}\psi_{n-1,n}(x_{n-1},x_n)\cdots\left\{\sum_{x_3}\psi_{3,4}(x_3,x_4)\underbrace{\left\{\underbrace{\sum_{x_2}\psi_{2,3}(x_2,x_3)\underbrace{\left\{\sum_{x_1}\psi_{1,2}(x_1,x_2)\right\}}_{\text{K sums of K values}}}_{\text{K evaluations of K products}}\right\}}_{\text{K sums of K values}}\right\}\cdots\right\}$$

The cost for computing the part shown in orange is $O(K^2)$.

Review

Message passing interpretation

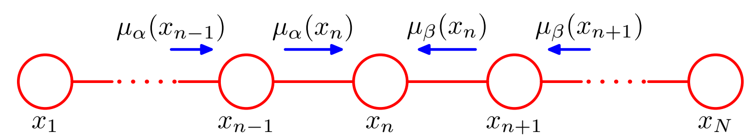


Define $\mu_\alpha(x_n)$ as a message passed from node x_{n-1} to node x_n.
Define $\mu_b(x_n)$ as a message passed from node x_{n+1} to node x_n.

Passing messages will correspond to the computation of taking input messages, and computing output messages.

Review

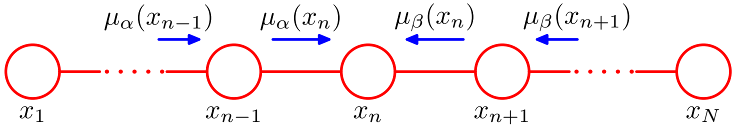
Message passing interpretation



$$\sum_{x_{n+1}}\cdots\sum_{x_{N-1}}\sum_{x_N}\prod_{i=n}^{N-1}\psi_{i,i+1}(x_i,x_{i+1})=$$

$$\underbrace{\left\{\sum_{x_{n+1}}\psi_{n,n+1}(x_n,x_{n+1})\cdots\underbrace{\left\{\sum_{x_{N-1}}\psi_{N-2,N-1}(x_{N-2},x_{N-1})\underbrace{\left\{\sum_{x_N}\psi_{N-1,N}(x_{N-1},x_N)\right\}}_{\mu_b(x_{N-1})}\right\}}_{\mu_b(x_{N-2})}\cdots\right\}}_{\mu_b(x_n)}$$

Message passing interpretation

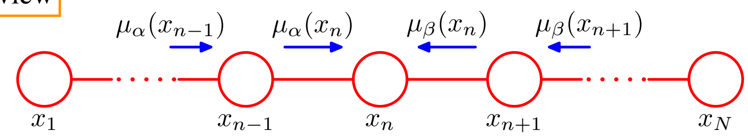


$$p(x_n) = \frac{1}{Z}\mu_a(x_n)\mu_b(x_n)$$

Algorithm

Send a message from x_1 to x_n.
Send a message from x_N to x_n.
Element wise multiply messages.
Normalize by summing over values of x_n (Z).

Review



To compute all marginals, send a message from left to right, and right to left, storing the result. Now compute any marginal as before.

If a node is observed, then we do the obvious. Specifically, we clamp the values of variables to the particular case.

This means that messages flowing into an observed node do not affect messages flowing out, as these are set to the "clamped" value.

Factor Graphs

Suppose p(**x**) factorizes as:

$$p(\mathbf{x}) = \prod_s f(x_s) \qquad \text{where } x_s \text{ are sets of of variables within } \mathbf{x}.$$

Make a node for each x_i as usual.

Now, make a different kind of node for *f()* (e.g., squares).

Draw edges between the factor nodes and the variables in the variable set, *s*.

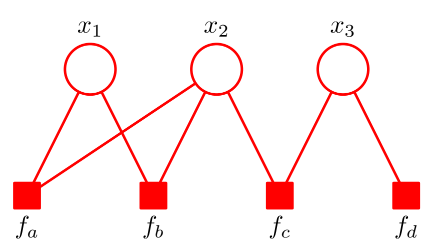
Note that the factorization formula means that we can convert **both** directed and undirected graphs to factor graphs.

Factor Graph Example

Suppose p(**x**) factorizes as:

$$p(\mathbf{x}) = \prod_s f(x_s) = f_a(x_1,x_2)f_b(x_1,x_2)f_c(x_2,x_3)f_d(x_3)$$

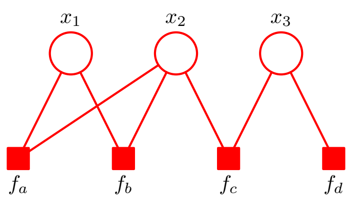
The graph is:



Factor Graph Example (continued)

Suppose p(**x**) factorizes as:

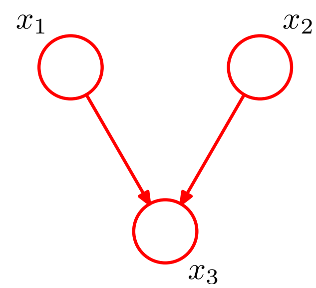
$$p(\mathbf{x}) = \prod_s f(x_s) = f_a(x_1,x_2) f_b(x_1,x_2) f_c(x_2,x_3) f_d(x_3)$$

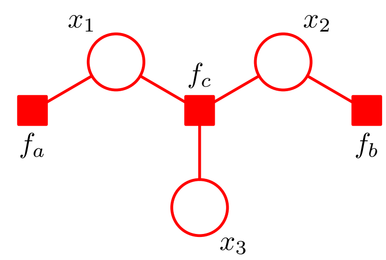


This layout emphasizes that factor graphs are *bipartite*.

Note two factors for the clique for 1 and 2, suggesting that factor graphs can preserve extra structure compared to undirected graphs.

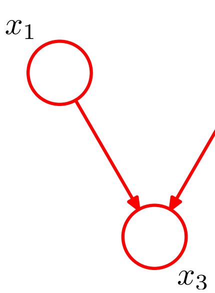
Factor Graph Example (2)

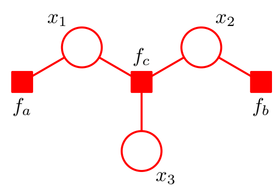




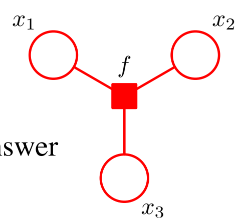
$$p(\mathbf{x}) = \underbrace{p(x_1)}_{f_a}\underbrace{p(x_2)}_{f_b}\underbrace{p(x_3|x_1,x_2)}_{f_c}$$

Factor Graph Example (2)





second answer



Factor Graph Summary

$p(\mathbf{x}) = \prod_s f(x_s)$ where x_s are sets of of variables within **x**.

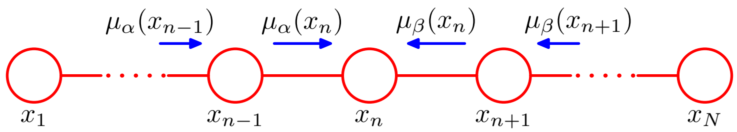
Denote variables by circles

Denote each factor by a square

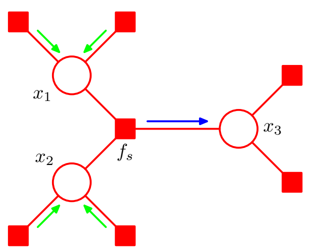
Draw links between squares and variables in the sets x_s.

Factor graphs are bipartite

Factor graph for a distribution is not necessarily unique.



Factor graphs conveniently represent the extended message passing needed for inference on trees/polytrees.

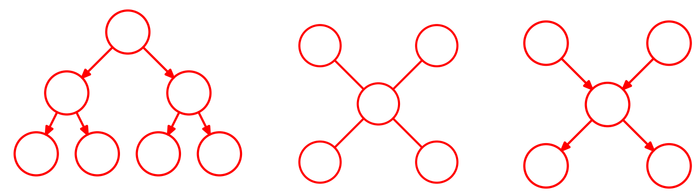


Trees/Polytrees

A directed graph is tree if the root node has no parents, others have exactly one parent.

An undirected graph is a tree if there is only one path between any pair of nodes.

A directed graph is a polytree if there is only one path per pair of nodes.



Factor Graphs and Trees

The factor graphs for directed trees, undirected trees, and directed polytrees are all trees.

(Recall definition for undirected trees---there is only one path between any two nodes).

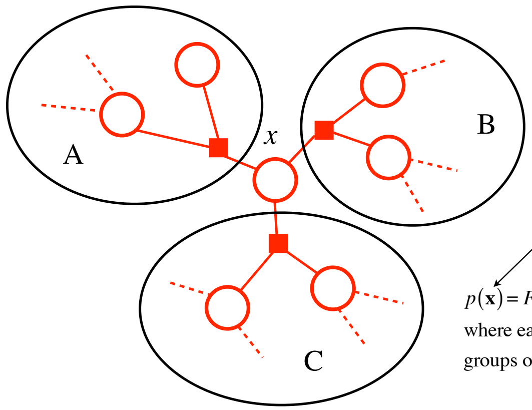
This means that (variable) node, x, with K branches divides a tree into K subtrees whose factors do not share variables except x.

Sum-product algorithm

Generalizes what we did with chains.

Generalizes and simplifies an algorithm introduced as “belief propagation”.

As with chains, consider the problem of computing the marginal of a selected node, x.



x connects subgraphs with node sets A, B, C.

vector (all vars)

$p(\mathbf{x}) = F(x,X_A)F(x,X_B)F(x,X_C)$

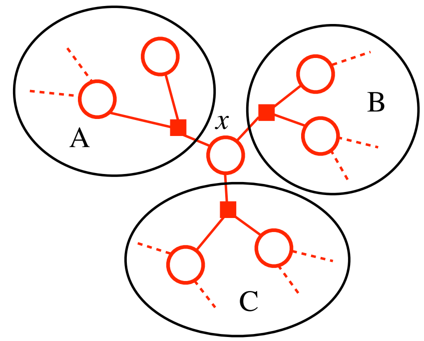
where each of these three factors are themselves groups of factors over x and the subgraphs.

More explicitly,

$F(x,X_A) = \prod_s f(X_s)$ with $X_s \subseteq \{x\} \cup A$

$F(x,X_B) = \prod_s f(X_s)$ with $X_s \subseteq \{x\} \cup B$

$F(x,X_C) = \prod_s f(X_s)$ with $X_s \subseteq \{x\} \cup C$



$p(\mathbf{x}) = F(x,X_A)F(x,X_B)F(x,X_C)$

where each of these three factors are themselves groups of factors over x and the subgraphs.

More explicitly,

$F(x,X_A) = \prod_s f(X_s)$ with $X_s \subseteq \{x\} \cup A$

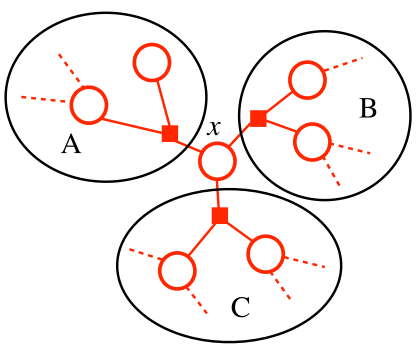
$F(x,X_B) = \prod_s f(X_s)$ with $X_s \subseteq \{x\} \cup B$

$F(x,X_C) = \prod_s f(X_s)$ with $X_s \subseteq \{x\} \cup C$

$$p(x) = \sum_{X \setminus \{x\}} p(\mathbf{x}) = \sum_{X \setminus \{x\}} F(x,X_A)F(x,X_B)F(x,X_C) = \left(\sum_A F(x,X_A)\right)\left(\sum_B F(x,X_B)\right)\left(\sum_C F(x,X_C)\right)$$

(recall our fancy formula)

$$\left(\sum a_i\right)\left(\sum b_j\right) = \sum\sum a_i b_j$$



$p(\mathbf{x}) = F(x,X_A)F(x,X_B)F(x,X_C)$

where each of these three factors are themselves groups of factors over x and the subgraphs.

More explicitly,

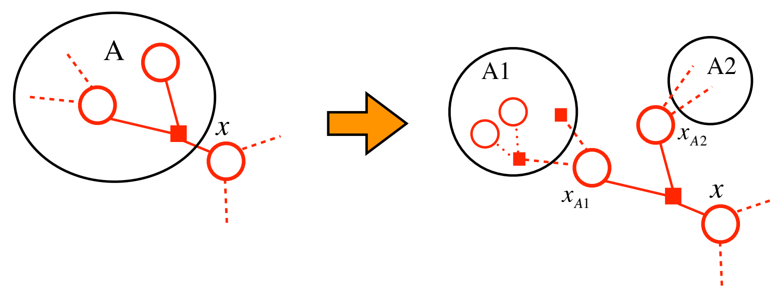
$F(x,X_A) = \prod_s f(X_s)$ with $X_s \subseteq \{x\} \cup A$

$F(x,X_B) = \prod_s f(X_s)$ with $X_s \subseteq \{x\} \cup B$

$F(x,X_C) = \prod_s f(X_s)$ with $X_s \subseteq \{x\} \cup C$

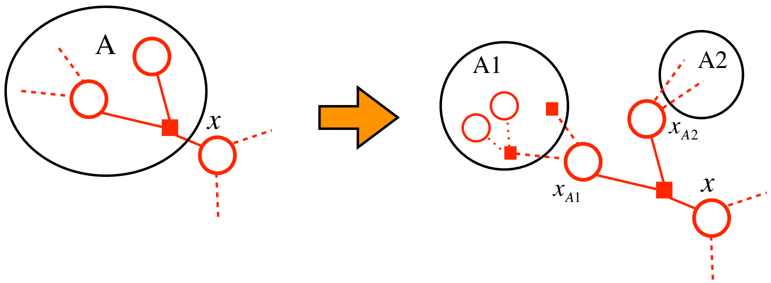
$$p(x) = \sum_{X \setminus \{x\}} p(\mathbf{x}) = \sum_{X \setminus \{x\}} F(x,X_A)F(x,X_B)F(x,X_C) = \left(\sum_A F(x,X_A)\right)\left(\sum_B F(x,X_B)\right)\left(\sum_C F(x,X_C)\right)$$

Consider the first one



Considering the first factor in the product on the previous slide,

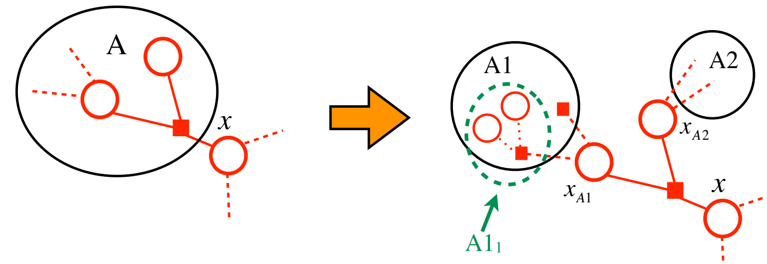
$$\sum_A F(x,X_A) = \sum_A f(x,x_{A1},x_{A2})F_{A1}(x_{A1},A1)F_{A2}(x_{A2},A2)$$
$$= \sum_{x_{A1},x_{A2}} f(x,x_{A1},x_{A2}) \sum_{A1} F_{A1}(x_{A1},A1) \sum_{A2} F_{A2}(x_{A2},A2)$$



$$\sum_A F(x,X_A) = \sum_A f(x,x_{A1},x_{A2}) F_{A1}(x_{A1},A1) F_{A2}(x_{A2},A2)$$

$$= \sum_{x_{A1},x_{A2}} f(x,x_{A1},x_{A2}) \sum_{A1} F_{A1}(x_{A1},A1) \sum_{A2} F_{A2}(x_{A2},A2)$$

To continue the expansion, consider the first one



$$\sum_A F(x,X_A) = \sum_A f(x,x_{A1},x_{A2}) F_{A1}(x_{A1},A1) F_{A2}(x_{A2},A2)$$

$$= \sum_{x_{A1},x_{A2}} f(x,x_{A1},x_{A2}) \sum_{A1} F_{A1}(x_{A1},A1) \sum_{A2} F_{A2}(x_{A2},A2)$$

This expands to

$$F_{A1}(x_{A1},A1) = \prod_{ne(x_{A1})\setminus f_{x,A}} F_{A1,i}(x_{A1},A1_i)$$

Notice that the sets A and $\mathbf{A1_i}$ have the same structure.

Sum-product algorithm

We could continue on recursively until we get to the leaf nodes, thereby computing *p(x)* via recursion.

However, a message passing implementation is simpler, and is better suited to computing all marginals at once.

Observations about factor graphs for trees

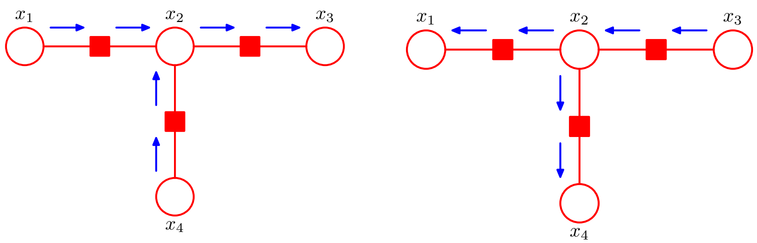
Any node can be root

Any node with K links splits the graph into K subgraphs which do not share nodes.

If we pass messages from:
1) the leaves to a chosen root;
2) the chosen root to the leaves,
then **all** messages that **can** be passed **have** been passed.

Further, the number of messages in 1 and 2 are the same.

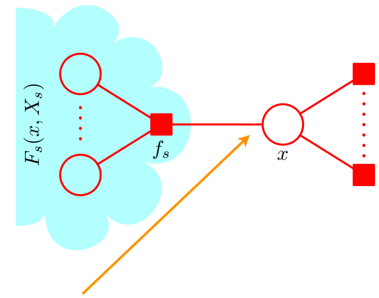
Observations about factor graphs



(x_3 is the root)

Sum-product algorithm

We defined two kinds of messages:
1) From factors to nodes.
2) From nodes to factors.



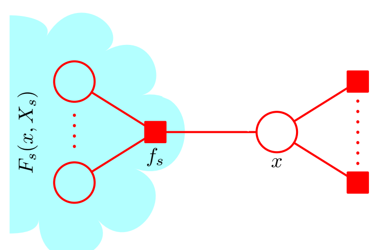
In analogy with chains, factor-to-node messages provide marginal distributions for a subgraph with the node. (In the chain case, we had the left side and the right side).

In the chain case we did not have factor nodes. This worked because the second kind of message (nodes-to-factor) is just "pass through" or "copy" in the case of only two links. So, we described it as simply passing messages from node to node.

Marginal distribution for a node x

$$p(x) = \sum_{\mathbf{x}/x} \prod_{s \in n(x)} F(x, X_s)$$

(marginal distribution for a node, x)



Marginal distribution for a node x

$$p(x) = \sum_{\mathbf{x}/x} \prod_{s \in n(x)} F(x, X_s)$$ (marginalize)

$$= \prod_{s \in n(x)} \left\{ \sum_{X_s} F(x, X_s) \right\}$$ (interchange sums and products)

(recall our fancy formula)

$$\left(\sum a_i\right)\left(\sum b_j\right) = \sum \sum a_i b_j$$

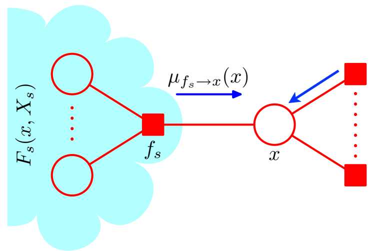
Note that each sum is simpler than what we started with because the variable sets are disjoint except for x.

Factor → node messages

$$p(x) = \sum_{\mathbf{x}/x} \prod_{s \in n(x)} F(x, X_s)$$

$$= \prod_{s \in n(x)} \left\{ \sum_{X_s} F(x, X_s) \right\}$$

$$= \prod_{s \in n(x)} \mu_{f_s \to x}(x)$$

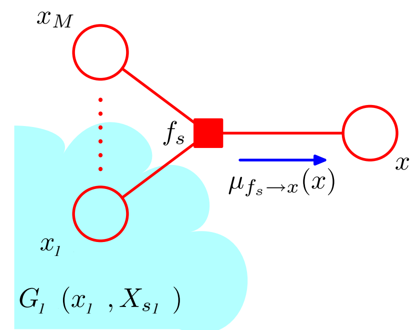


$$\mu_{f_s \to x}(x) \equiv \sum_{X_s} F(x, X_s)$$ (factor-to-node message)

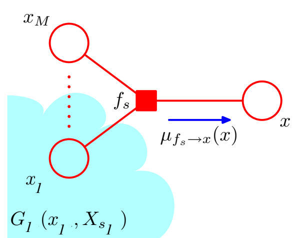
Computing the factor → node messages

$$\mu_{f_s \to x}(x) \equiv \sum_{X_s} F_s(x, X_s)$$ (sum removes all variables except x.)

Where $F_s(x, X_s) = f_s(x, x_1, x_2, \ldots, x_M)\underbrace{G_1(x_1, X_{s1})G_2(x_2, X_{s2})\ldots G_M(x_M, X_{sM})}_{\text{Collections of factors in the M sub-graphs}}$



Computing the factor → node messages



$$\mu_{f_s \to x}(x) \equiv \sum_{X_s} F_s(x, X_s)$$ (sum removes all variables except x.)

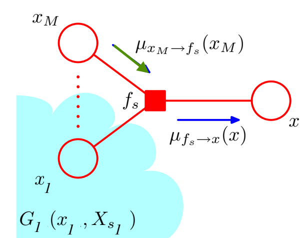
Where $F_s(x, X_s) = f_s(x, x_1, x_2, \ldots, x_M)G_1(x_1, X_{s1})G_2(x_2, X_{s2})\ldots G_M(x_M, X_{sM})$

$$\sum_{X_s} F_s(x, X_s) = \sum_{x_1}\sum_{x_2}\cdots\sum_{x_M} f_s(x, x_1, x_2, \ldots, x_M) \sum_{X_{s1}}\sum_{X_{s2}}\cdots\sum_{X_{sM}} G_1(x_1, X_{s1})G_2(x_2, X_{s2}) \;....\; G_M(x_M, X_{sM})$$

So, $\sum_{X_s} F_s(x, X_s) = \sum_{x_1}\sum_{x_2}\cdots\sum_{x_M} f_s(x, x_1, x_2, \ldots, x_M) \prod_{m \in ne(f_s)\backslash x} \sum_{X_{sm}} G_m(x_m, X_{sm})$

(interchanging sums and products)

Computing the factor → node messages



$$\sum_{X_s} F_s(x, X_s) = \sum_{x_1}\sum_{x_2}\cdots\sum_{x_M} f_s(x, x_1, x_2, \ldots, x_M) \prod_{m \in ne(f_s)\backslash x} \sum_{X_{sm}} G_m(x_m, X_{sm})$$

$$= \sum_{x_1}\sum_{x_2}\cdots\sum_{x_M} f_s(x, x_1, x_2, \ldots, x_M) \prod_{m \in ne(f_s)\backslash x} \mu_{x_m \to f_s}(x_m)$$

where we define:

$$\mu_{x_m \to f_s}(x_m) \equiv \sum_{X_{sm}} G_m(x_m, X_{sm})$$ (node → factor messages)