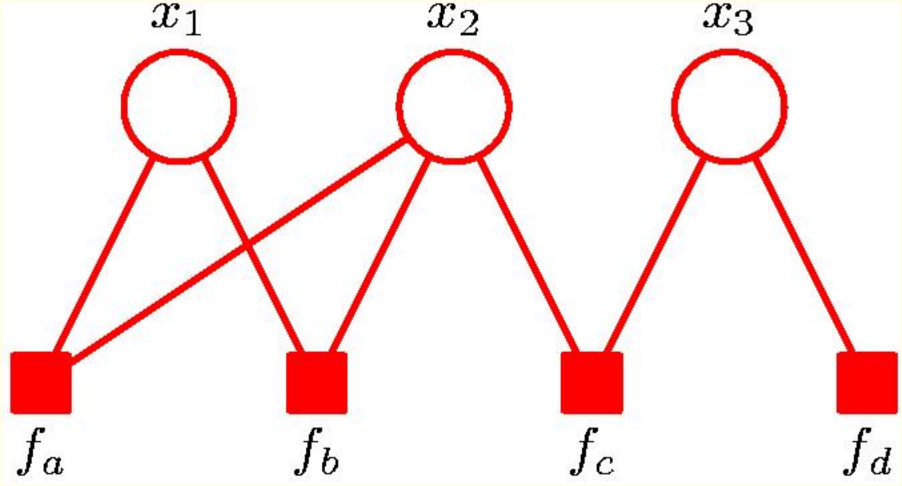
CSC2535 Spring 2013

Lecture 2a: Inference in factor graphs

Geoffrey Hinton

Factor graphs: A better graphical representation for undirected models with higher-order factors

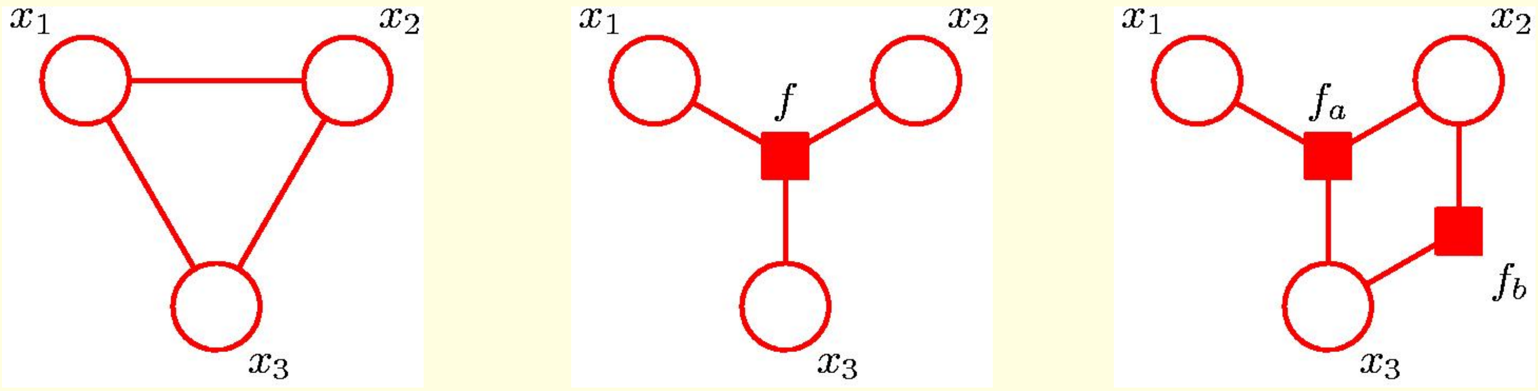


$$p(\mathbf{x}) = \prod_s f_s(\mathbf{x}_s)$$

If the potentials are not normalized we need an extra factor of 1/Z.

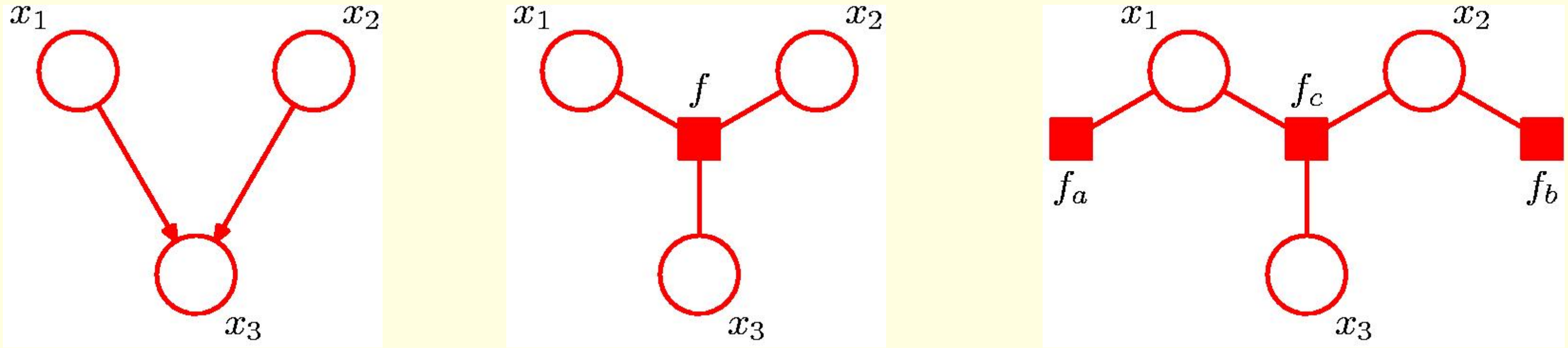
- Each potential has its own factor node that is connected to all the terms in the potential.
- Factor graphs are always bipartite.

Representing a third-order term in an undirected model



- The third-order factor is much more visually apparent than the clique of size 3.
- It easy to divide a factor into the product of several simpler factors.
  - This allows additional factorization to be represented.

Converting trees to factor graphs



- When we convert any singly connected graphical model to a factor graph, it remains singly connected.
  - This preserves the simplicity of inference.
- Converting a singly connected directed graph to an undirected graph may not preserve the property of being singly connected.

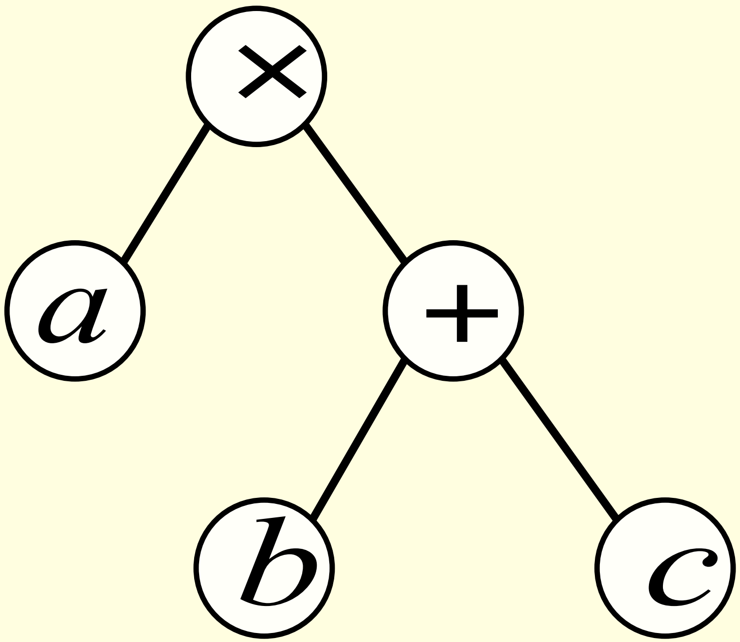
Computing a marginal in a factor graph with nodes that have discrete values

$$p(x_n) = \sum_{\mathbf{x} \setminus x_n} p(\mathbf{x})$$

- To obtain the marginal probability function for x_n we could consider each possible value of x_n and sum the joint probability over all possible values of all the other random variables.
  - This would take a time that was exponential in the number of other variables.

Expression trees

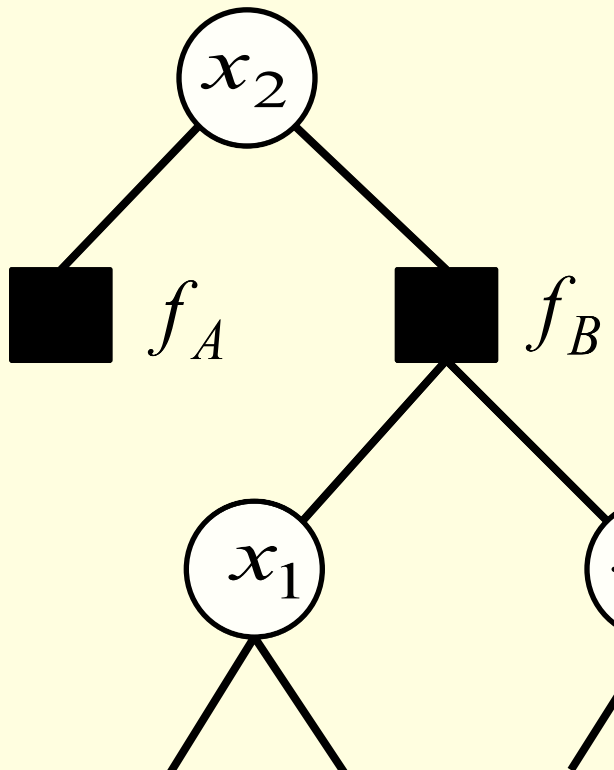
$$ab + ac = a(b + c)$$

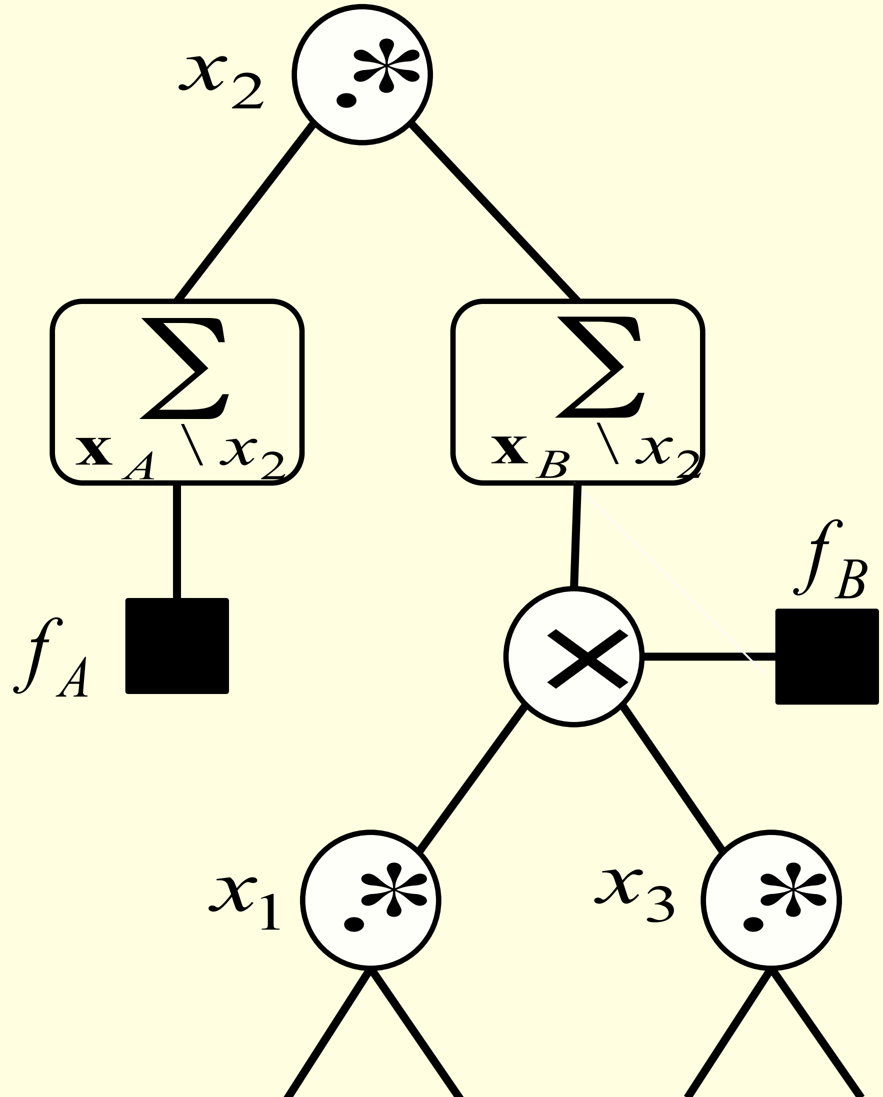


- We can compute the values of arithmetic expressions in a tree.
- We can do the same thing using probability distributions instead of scalar values.
  - The product operation gets replaced by a pointwise product.
  - The sum operation get replaced by something more complicated.

Converting a factor graph to an expression tree

To compute a marginal, the factor graph is drawn with the variable of interest at the top.





The messages passed up the tree

- A message is a function that specifies how much it likes each of the possible values of a variable.
  - How much it likes the value is a *probability.
- The message from a variable to a factor is the product of the messages the variable receives from the factors below it.
  - So its a function over the values of the sending variable. It summarizes the relevant aspects of the combined opinions of all the stuff below that variable.
- The message from a factor to a variable is more complicated.
  - It is a function over the values of the receiving variable. It summarizes the relevant aspects of the combined opinions of all the stuff below that factor.

The message from a factor to a variable

- A factor can see the vector of *probabilities for each of the variables below it. It needs to convert these vectors into a vector of *probabilities for the variable above it.
- For each combination of values of the variables below it, the factor node does the following:
  - First it computes the product, P, of the *probabilities that the variables below have for that combination.
  - Then, for each value of the variable above, it multiplies P by the value of the factor to get a function over the values of the variable above it.
- Finally, the factor node adds up these functions over all possible combinations of values of the variables below it.

The messages in math

message

$$\mu_{x_m \to f_s}(x_m) = \prod_{s' \in \mathrm{ne}(x_m) \setminus f_s} \mu_{f_{s'} \to x_m}(x_m)$$

variable

factor

factors below

$$\mu_{f_s \to x_m}(x_m) = \sum_{\mathbf{x}_s \setminus \mathrm{x}_\mathrm{m}} \left(f_s(\mathbf{x}_s) \prod_{m' \in \mathrm{ne}(f_s)} \mu_{x_{m'} \to f_s}(\mathbf{x}_{m'}) \right)$$

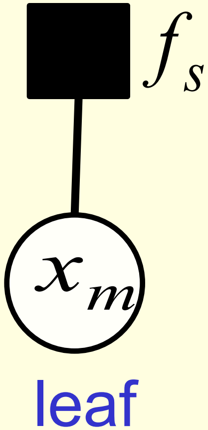
sum over all combinations of values of variables below

variables below

The messages at the leaf nodes

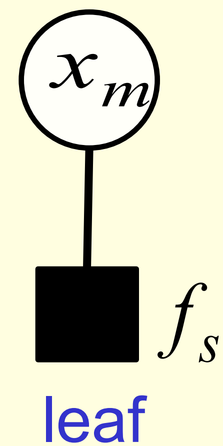
- For a variable that is only connected to one factor:

$$\mu_{x_m \to f_s}(x_m) = 1$$



- For a factor that is only connected to one variable:

$$\mu_{f_s \to x_m}(x_m) = f_s(x_m)$$



Starting and finishing: Method 1(only for trees)

- Start by sending messages from all the leaf variables and factors.
- Then send a message whenever all of the messages it depends on are present.
- To get the marginals for all variables, allow messages to flow in both directions.

$$p^*(x_m) = \prod_{s \in \mathrm{ne}(x_m)} \mu_{f_s -> x_m}(x_m), \qquad Z = \sum_{x_m} p^*(x_m)$$

Starting and finishing: Method 2 (works with loops)

- Start by sending a message of 1 from *every* variable (not just the leaf ones).
- Then compute messages as normal.
- After a time equal to the diameter of the graph this will settle to the right answer (if the graph is singly connected).
  - It wastes a lot of computation if the graph is singly connected.
- It often computes useful answers in loopy graphs!