CSC2535 Spring 2011

Lecture 1: Introduction to Machine Learning and Graphical Models

Geoffrey Hinton

How to represent a probability distribution over several random variables

- There are two different ways represent a distribution over several random variables:

$$p(X_1 = x_1, X_2 = x_2, X_3 = x_3, X_4 = x_4)$$

which we abbreviate as $p(x_1, x_2, x_3, x_4)$

- **Product of conditional probabilities**:

$$p(x_1, x_2, x_3, x_4) = p(x_4)\,p(x_3 \mid x_4)\,p(x_2 \mid x_3, x_4)\,p(x_1 \mid x_2, x_3, x_4)$$

- **Global energy function**:

$$p(x_1, x_2, x_3, x_4) = \frac{1}{Z} e^{-E(x_1, x_2, x_3, x_4)}$$

Disdvantages and advantages of the energy-based approach

- To compute the probability of a joint configuration we need to know the partition function, Z.
  - Z has exponentially many terms (for discrete variables)
- To change the the parameters of the energy function so as to improve the probability of the training data, we need the derivative of Z with respect to each parameter.
  - The exact derivative requires exponential work.
- We can define the energy of a joint configuration of the variables in almost any way we like and we will still get a proper distribution
  - But it must integrate to less than infinity over all joint configurations.

Less general distributions over several random variables

- The simplest distribution is when the variables do not interact at all:

  $$p(x_1, x_2, x_3, x_4) = p(x_1)p(x_2)p(x_3)p(x_4)$$

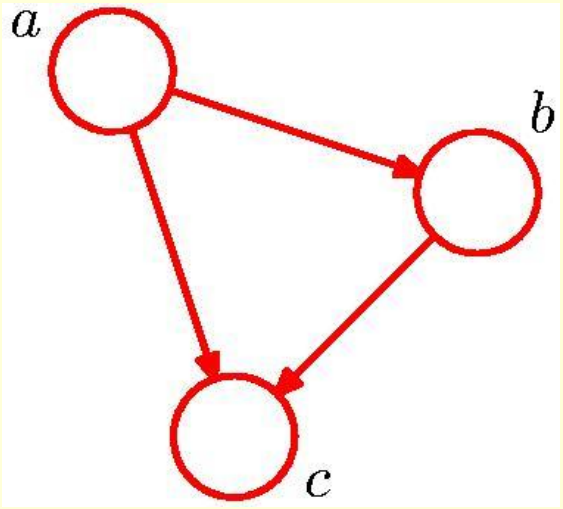
  This is called a factorial distribution.

  There are many other ways to represent a distribution using a product of conditional distributions or a sum of local energies that are more complicated than complete independence, but less complicated than fully a general distribution. This is what Graphical Models is all about.

Three types of graphical model

- **Directed models** use conditional probabilities
  - Each conditional probability must be properly normalized.
- **Undirected models** use energy functions that are a sum of several terms.
  - The terms in the energy function are very flexible and each variable can be involved in many different terms without causing problems. But the partition function is nasty.
- **Hybrid models** (like a "deep belief net") combine directed and undirected pieces.

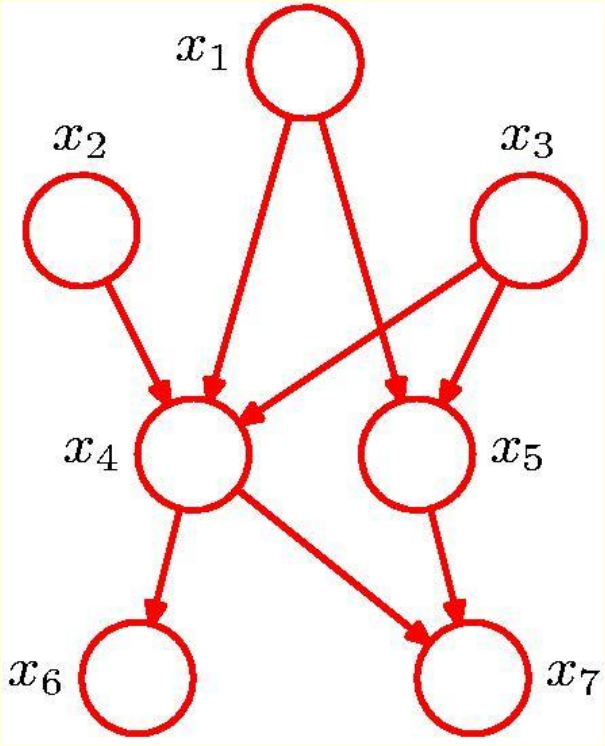
A graphical representation of a set of conditional probabilities



- Each node represents a random variable.
- Each directed edge represents an explicit dependency on a “parent”
- For general distributions, the graph is fully connected.

$$p(a,b,c) = p(c \mid a,b)\, p(b \mid a)\, p(a)$$

Representing less general distributions



- The structure of a less general distribution can be represented by the missing edges.
- If the directed graph is acyclic and the distribution of each node conditional on its parents is normalized, the whole distribution will be consistent .

$$p(\mathbf{x}) = \prod_k p(x_k \mid pa_k) = p(x_1)\,p(x_2)\,p(x_3)\,p(x_4 \mid x_1, x_2, x_3)\ldots$$

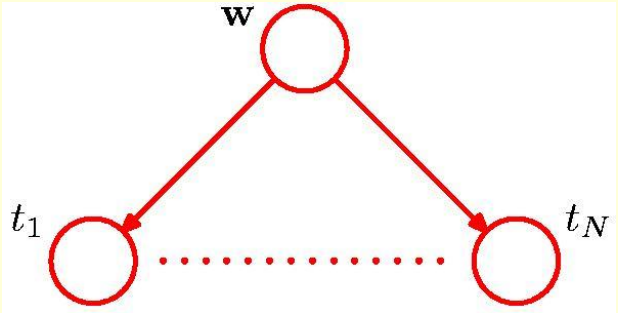
parents

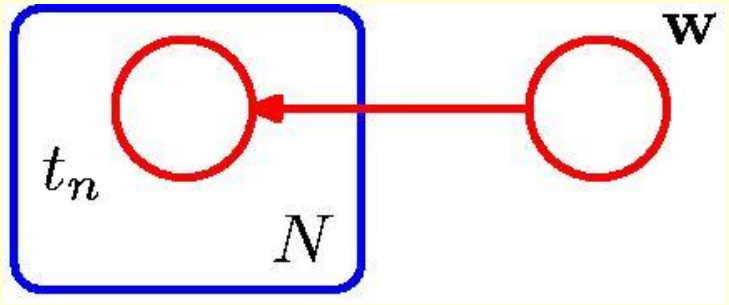
$$p(x_5 \mid x_1, x_3)\,p(x_6 \mid x_4)\,p(x_7 \mid x_4, x_5)$$

Bayesian polynomial regression

$$p(t \mid x, \mathbf{x}, \mathbf{t}) = \int p(t \mid x, \mathbf{w})\, p(\mathbf{w} \mid \mathbf{x}, \mathbf{t})\, d\mathbf{w}$$

test train

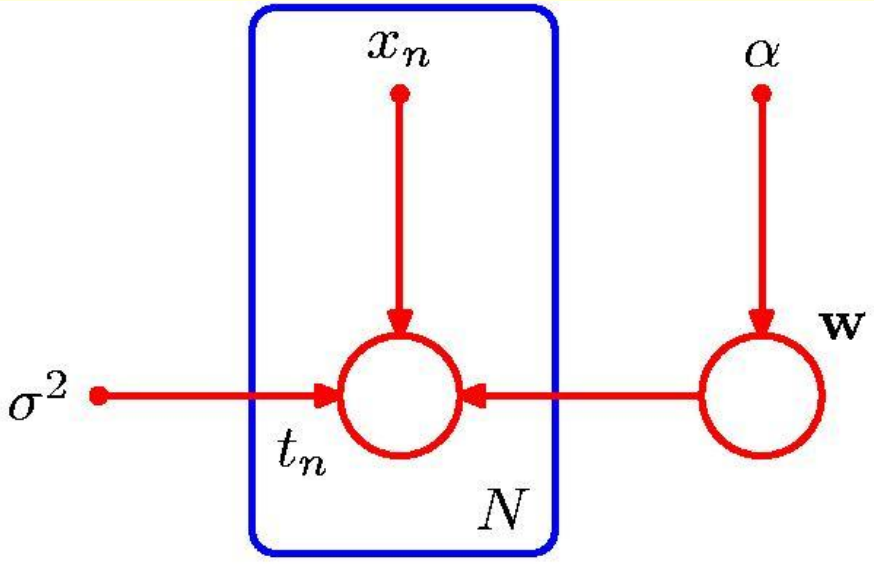




- The modeled random variables are **t** and **w**
- The inputs, **x**, are given. They are not random variables in the model.
- The "plate" notation is used for multiple variables with the same dependencies.

$$p(\mathbf{t}, \mathbf{w}) = p(\mathbf{w}) \prod_{n=1}^{N} p(t_n \mid \mathbf{w})$$

Showing dependencies on deterministic parameters

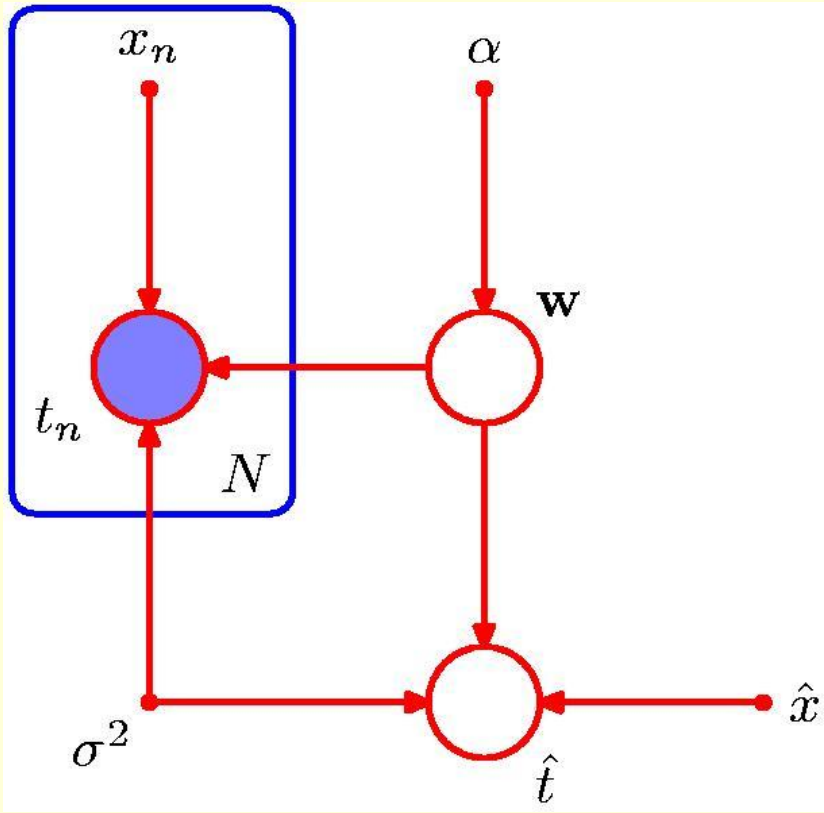


- We can use a small solid circle for a parameter such as:
  - Output noise variance
  - Input vector for a case
  - Parameter determining the prior distribution of the weights.

$$p(\mathbf{t}, \mathbf{w} \mid x, \alpha, \sigma^2) = p(\mathbf{w} \mid \alpha) \prod_{n=1}^{N} p(t_n \mid \mathbf{w}, x_n, \sigma^2)$$

A graphical model of a test prediction

- We represent the fact that a node has been observed by filling it in.
- The output noise variance affects both training and test data.

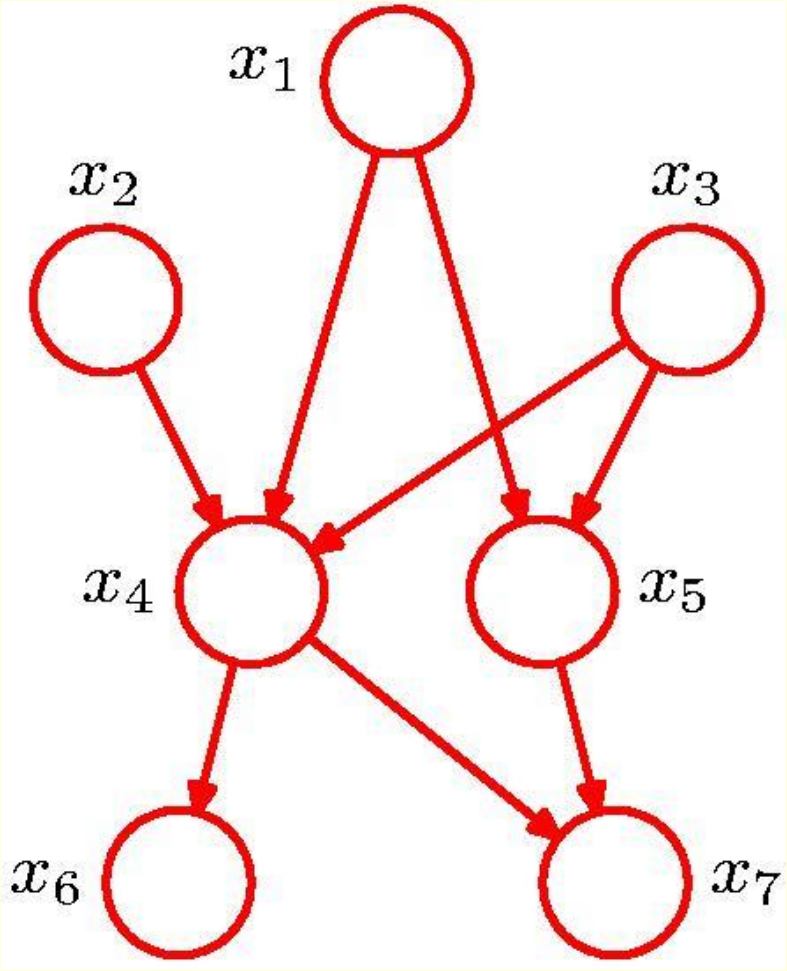


$$p(\hat{t}, \mathbf{t}, \mathbf{w} \mid \hat{x}, \mathbf{x}, \alpha, \sigma^2)$$

$$= p(\mathbf{w} \mid \alpha)\, p(\hat{t} \mid \hat{x}, \mathbf{w}, \sigma^2) \prod_{n=1}^{N} p(t_n \mid x_n, \mathbf{w}, \sigma^2)$$

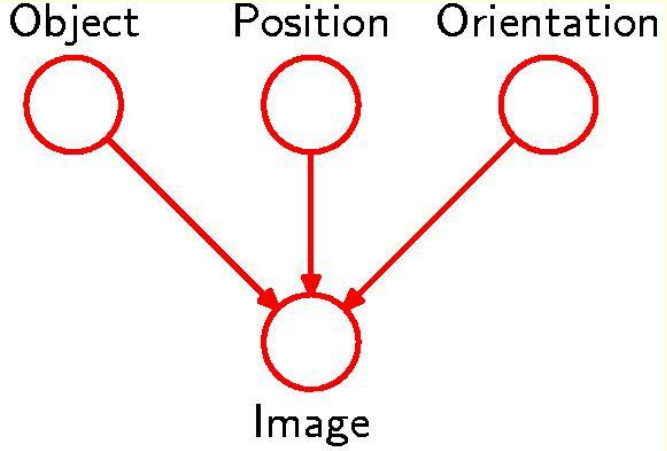
An important fact about acyclic directed graphical models

- An unobserved node has no effect on the distributions of its parents.
  - It only affects the distributions of its descendants.
  - The direction of the arrows is like time: Causes only affect the future.

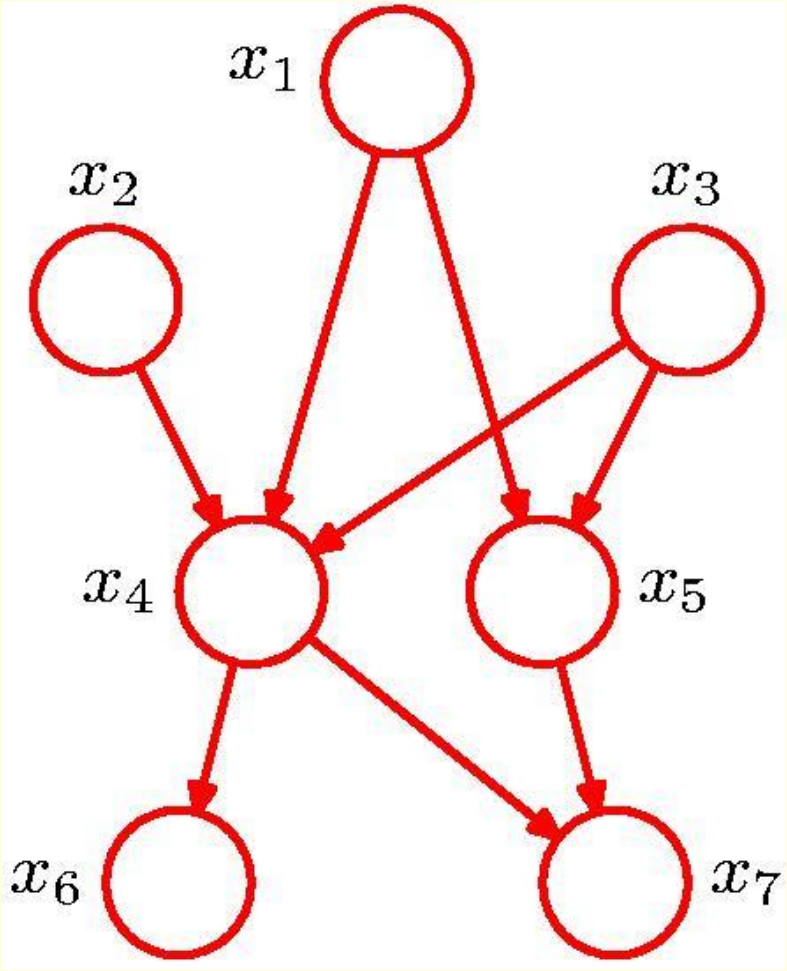


Ancestral sampling

- Start at the top and sample in order.
- Good for seeing what the model believes.



What false claims are made by this model?



Two very different approaches to directed graphical models

- We can view the higher-level nodes as unobserved causes that explain the statistical structure of the joint distribution over the observed variables.
  - Missing edges represent qualitative aspects of the statistical structure.
  - The individual conditional probability functions of the nodes represent quantitative aspects.
- We care a lot about where the edges are and we can interpret the individual nodes.
  - Graphical models evolved from expert systems.

Two very different approaches to directed graphical models (continued)

- Consider using small lego blocks to model the shape of a car. All we care about is the shape.
  - We do not really believe the car is made of lego.
  - The blocks are just “modeling stuff”. This stuff needs to be able to model any reasonable shape.
  - Its probably good if there are many different ways of modeling the same shape. “Identifiability” is not important.
- We can adopt a similar approach to modeling a complicated probability distribution.
  - The only role of the latent variables is to model the density (But with enough data the right model is best!).

An intermediate approach

- We are interested in the values of the latent variables, but we are not aiming for identifiability.
- We want to use the latent variables for tasks like object or speech recognition.
  - We expect the latent variables to be more directly related to classes we are interested in than the raw sensory inputs.
  - But there may be many different latent variable representations that are equally good.

Two very important types of random variable

- An analogy: If we start with integers, addition, subtraction and multiplication keep us in the domain of integers.
- If we start with discrete variables, inference keeps us in the domain of discrete variables.
- If we start with Gaussian variables, inference keeps us in the domain of Gaussian variables provided the conditional probability models are all linear.

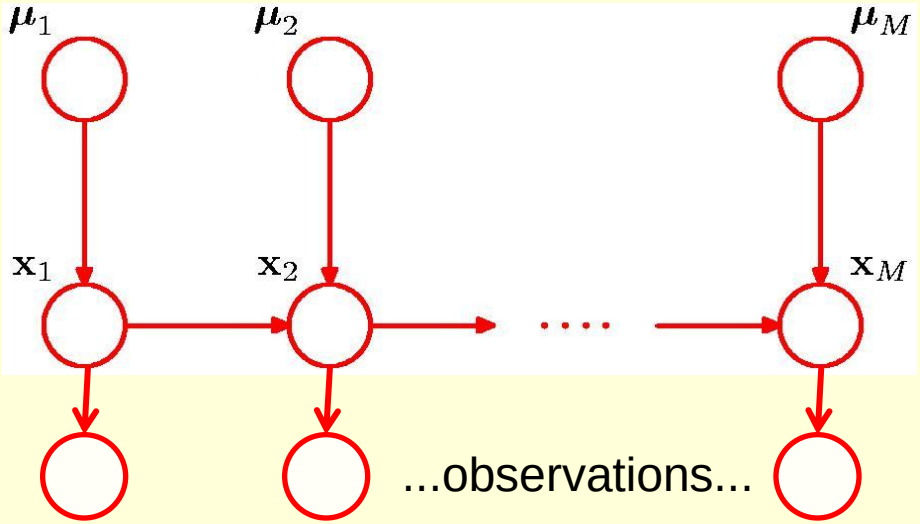
Reducing the number of parameters

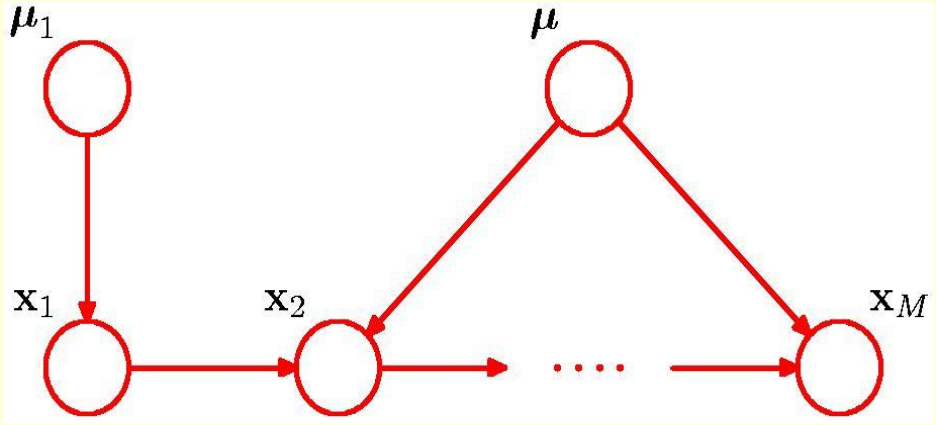
- For a chain of M nodes each with K states, instead of $K^M - 1$ we have $(K-1) + (M-1)K(K-1)$

start

- If the parameters are shared across time, we have: $(K-1) + K(K-1) = K^2 - 1$
- This is good for modeling stationary sequences.
  - It is the graphical model that forms the basis of a simple Hidden Markov Model.

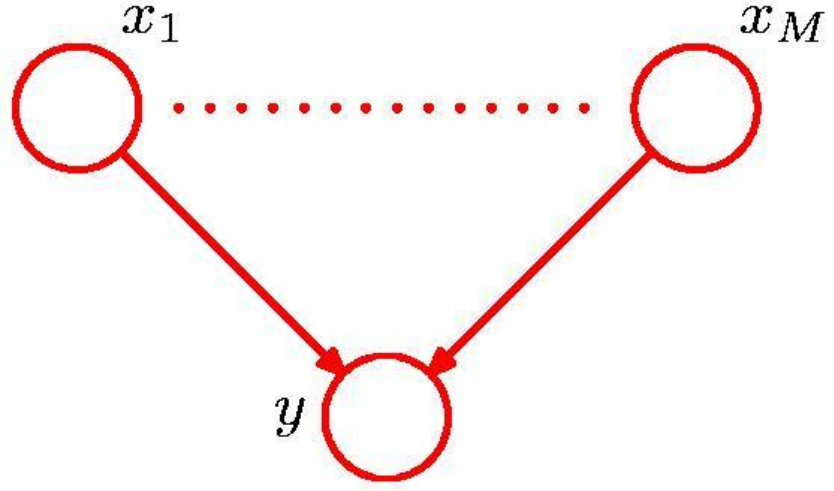
Adding priors to the graphical model of an HMM





- To be Bayesian, an HMM needs a prior over the parameters.
  - We can use a Dirichlet prior. This is conjugate. It is equivalent to having already seen some data.
- An HMM can share the prior over the transition parameters.

Replacing conditional probability tables by functions



A node with L states and M parents each with K states requires a table of size:

$$(L-1)\,K^M$$

- Suppose L=2
- We can use a logistic sigmoid function to reduce the number of parameters to M.
- This is a good idea if the logistic can approximate the table we want.

$$p(y=1\,|\,\mathbf{x}) = \sigma(\mathbf{w}^T\mathbf{x})$$

Graphical models with Gaussian random variables

- Engineers use these all the time, but people in AI hated real numbers and it took them a long time to go beyond discrete variables and look-up tables for the interactions.
- Replace the discrete distributions by Gaussian distributions and make the interactions linear:

$$p(x_i \mid \mathrm{pa}_i) = N\left(b_i + \sum_{j \,\varepsilon\, \mathrm{pa}_i} w_{ij} x_j, \quad v_i \right)$$

Gaussian mean variance

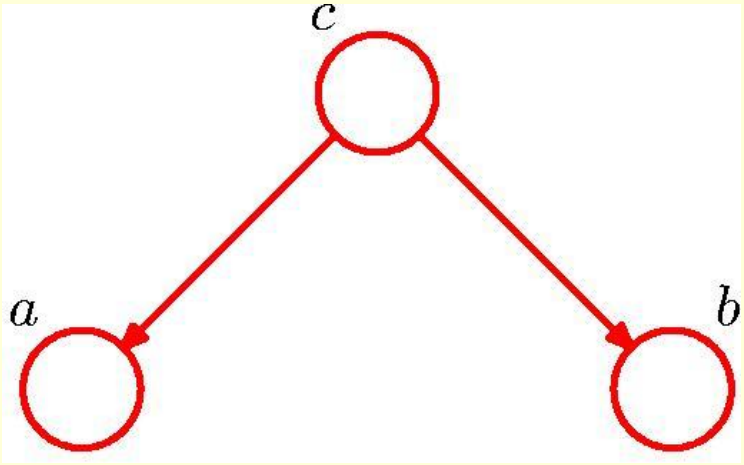
The joint distribution with Gaussian nodes

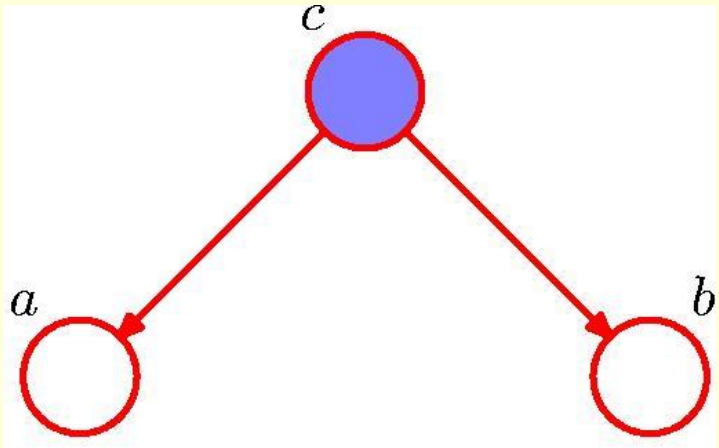
$$\ln p(\mathbf{x}) = \sum_{i=1}^{D} \ln p(x_i \mid \mathrm{pa}_i)$$

$$= -\sum_{i=1}^{D} \frac{1}{2v_i} \left(x_i - \left(b_i + \sum_{j \varepsilon \, \mathrm{pa}_i} w_{ij} x_j \right) \right)^2 + \mathrm{const}(\mathbf{v})$$

- Since the log prob is quadratic in **x,** the joint distribution is a multivariate Gaussian.
- We can determine the mean and covariance by using the symbolic equivalent of ancestral sampling:
  - Compute the mean and covariance of the Gaussian distribution for each node given the means and covariances of the distributions of its parents (see Bishop).

Conditional independence for tail-to-tail nodes





- If c has not been observed, a and b are, in general, not independent. They have a common cause.
- Once c has been observed, a and b can no longer have any effect on each other. They become independent.

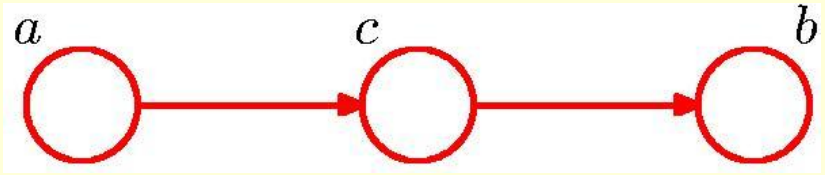
$$p(a \mid b, c) = p(a \mid c)$$

$$p(a, b \mid c) = p(a \mid c)\, p(b \mid c)$$

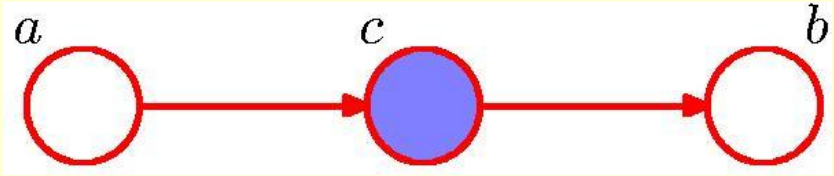
The importance of conditional independence

- Conditional independence makes inference much simpler.
  - The probability distributions over the values of a variable can be combined by pointwise multiplication if the are sources are independent.
- The graph structure can be used to read off the conditional independencies.

Conditional independence for head-to-tail nodes

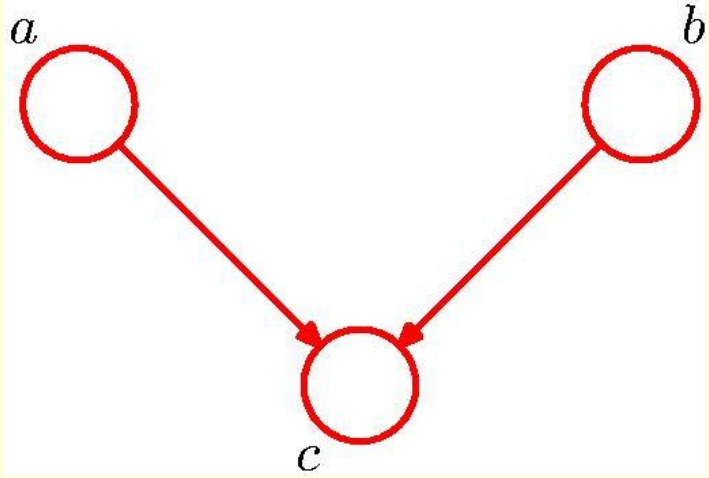


- If c is not observed, a can influence c and c can influence b, so $p(a,b) \neq p(a)p(b)$



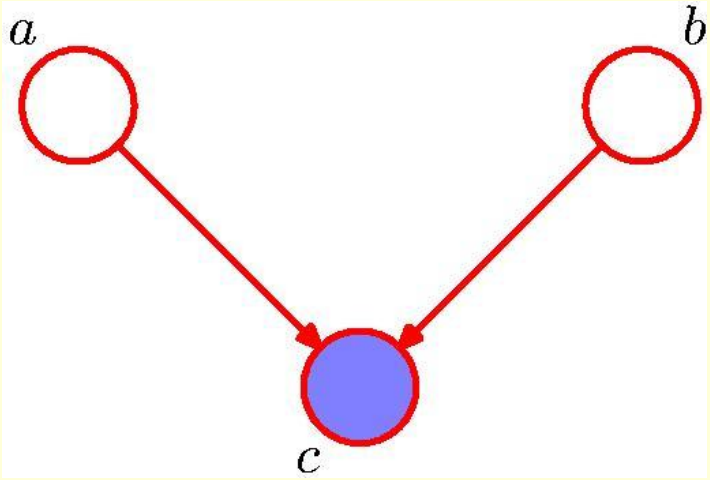
- If c is observed, the value of a can no longer influence it, so $p(a,b|c) = p(a|c)p(b|c)$

UNconditional independence for head-to-head nodes



- An unobserved descendant has no effect. So we have

$$p(a,b) = p(a)\,p(b)$$

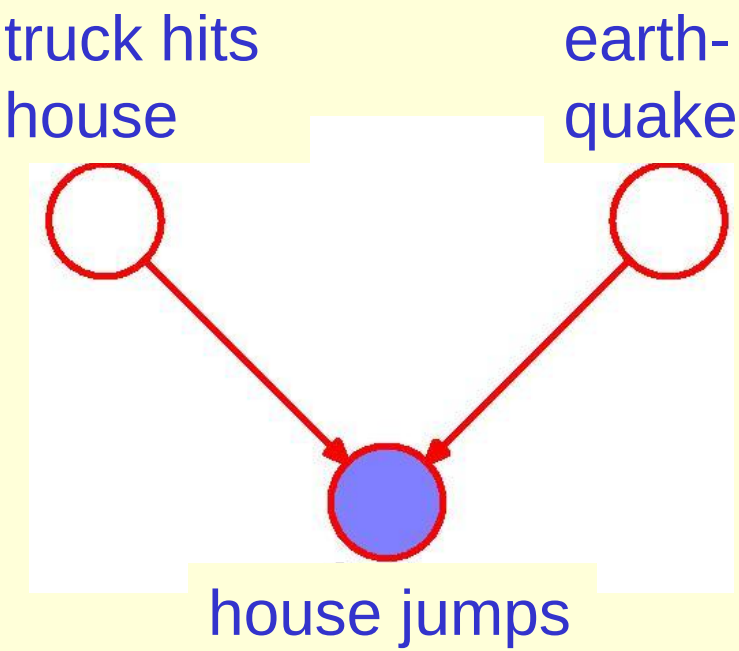


- If the descendant (or any of its descendants) is observed, its value has implications for both a and b, so

$$p(a,b \mid c) \neq p(a \mid c)\,p(b \mid c)$$

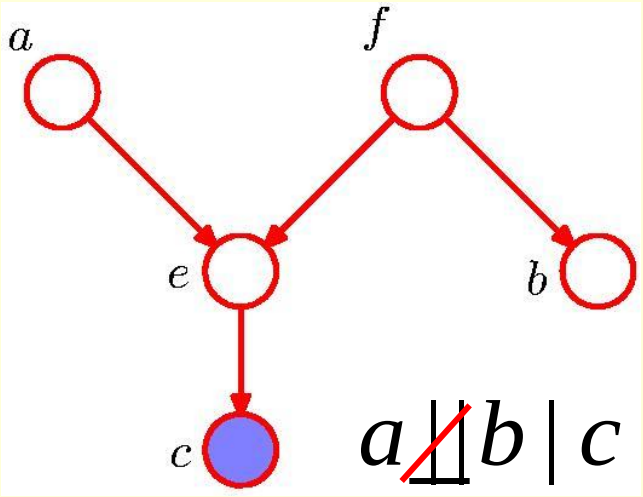
Explaining away

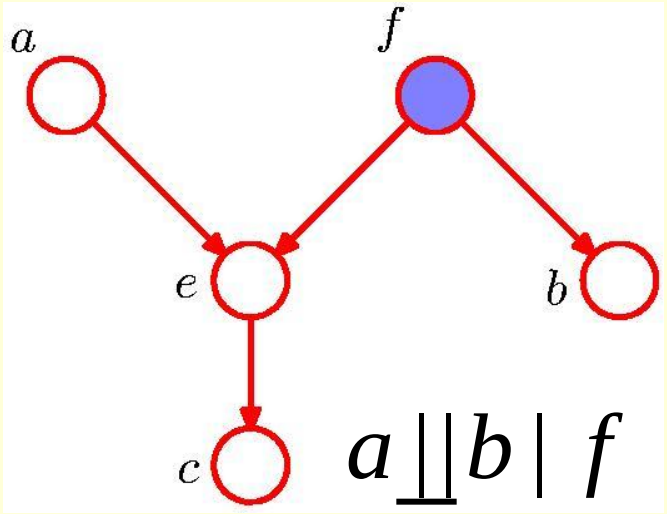
- Suppose that earthquakes are rare
- Suppose that trucks hitting houses is rare.
- Suppose that houses do not jump without a cause.
  - If you observe the house jumping, you need to assume that one of the causes happened.
  - One cause removes the need for the other cause.



The two causes are independent in the model, but anti-correlated after the observation.

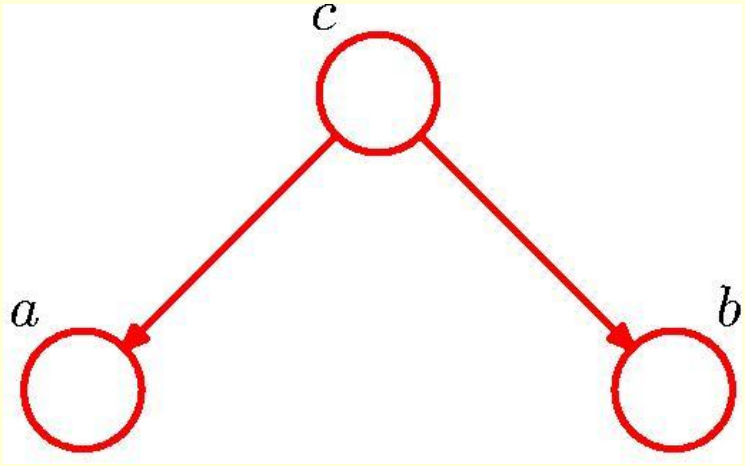
D-separation

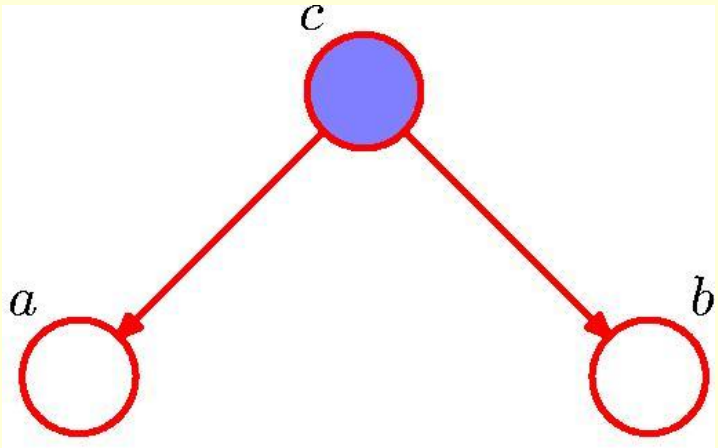




- a is independent of b if and only if all paths connecting a and b are blocked.
- head-to-tail and tail-to-tail nodes are blocked when observed.
- head-to-head nodes are blocked when the node and all its descendants are unobserved.

Naive Bayes and D-separation





- In this model, a and b are not independent when the class label c has not been observed.
- Once c is observed, a and b become independent. So for each particular class, it is easy to combine the effects of observing both a and b.

Combining observations in naive Bayes

$$p(a,b) \neq p(a)\,p(b)$$

$$p(a,b \mid c) = p(a \mid c)\,p(b \mid c)$$

- The conditional independence makes it easy to use Bayes theorem to combine evidence from multiple observations:

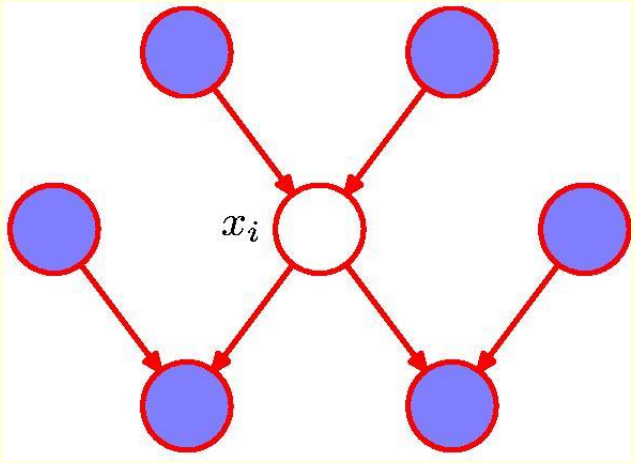
$$p(c \mid a,b) \propto p(c)\,p(a,b \mid c)$$

$$\propto p(c)\,p(a \mid c)\,p(b \mid c)$$

- Learning $p(a \mid c)$ is very easy because this distribution is only one-dimensional.

The Markov Blanket in a directed graphical model

- The Markov blanket of a node is the minimal set of nodes that must be observed to make this node independent of all other nodes.
- In a directed model, the blanket includes all the parents of the node's children.
  - This is because of explaining away.



Undirected graphical models
(Markov Random Fields, Energy-based models)

- The joint distribution over the random variables is defined to be proportional to the product of some potential functions defined over subsets of the variables:

$$p(\mathbf{x}) = \frac{1}{Z}\prod_{C}\varphi_C(\mathbf{x}_C), \qquad Z = \sum_{\mathbf{x}}\prod_{C}\varphi_C(\mathbf{x}_C)$$

- Equivalently, the joint distribution is defined via the sum of some energy functions which are each defined over subsets of the variables.

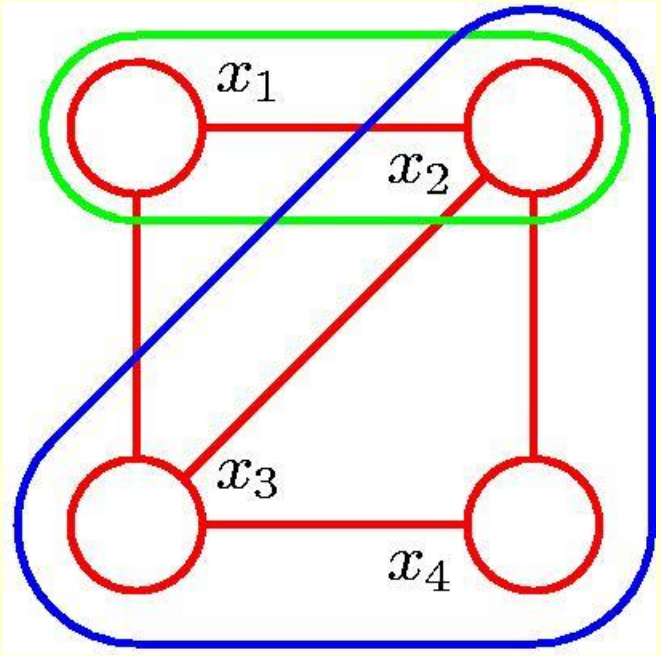
$$p(\mathbf{x}) = \frac{1}{Z}\exp\left(-\sum_{C}E(\mathbf{x}_c)\right), \quad where \;\; E(\mathbf{x}_C) = -\ln\varphi_C(\mathbf{x}_C)$$

Representing the relevant subsets

- The subsets that are used to define the potential functions (*i.e.* terms in the energy function) are represented by cliques in the undirected graph.

$$\{x_1, x_2\}, \{x_2, x_3\}, \{x_3, x_4\},$$
$$\{x_4, x_2\}, \{x_1, x_3\},$$
$$\{x_1, x_2, x_3\}, \{x_2, x_3, x_4\}$$

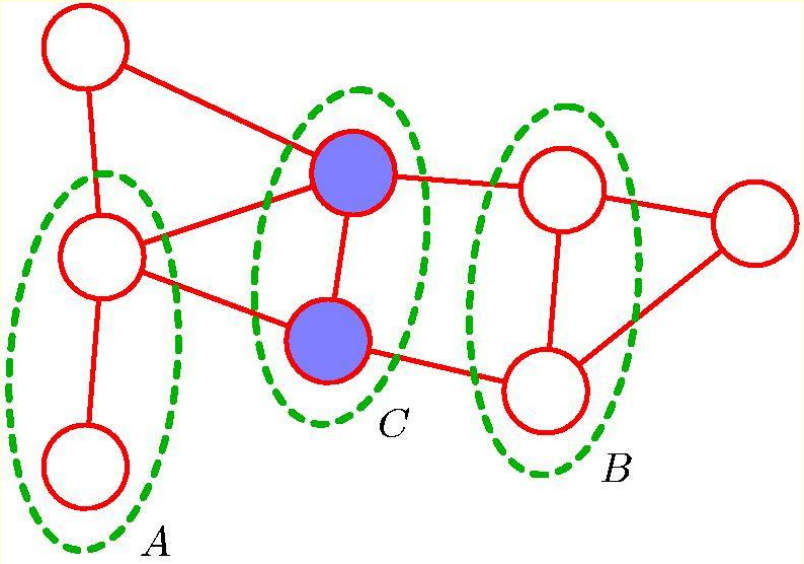
The cliques represented by this graph



Using cliques to represent factors

- If the factors (*i.e.* the potential functions or energy terms) only involve two nodes, an undirected graph is a nice representation.
- If the factors involve more than two nodes its not nearly such a nice representation.
  - A factor graph is a much nicer representation.

Conditional independence in an undirected model

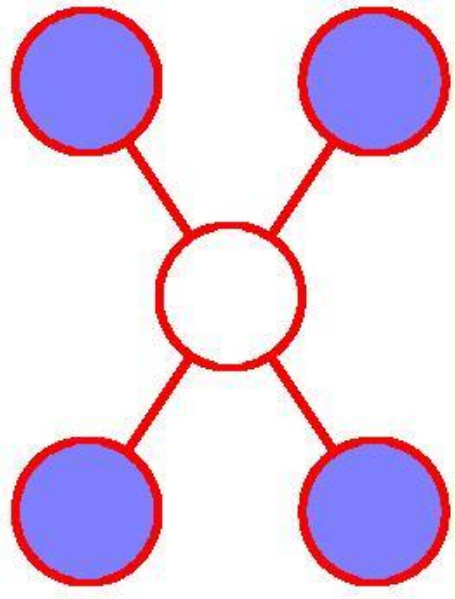


- This is easier than in a directed model.
  - Observation blocks a node.
  - Two sets of nodes are conditionally independent if the observations block all paths between them.

Conditional independence and factorization in undirected graphs

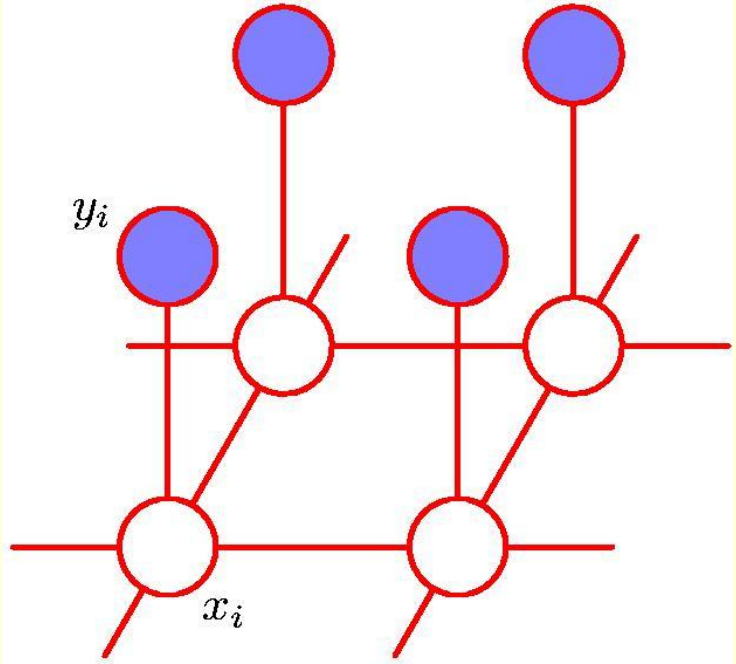
- Consider two sets of distributions:
  - The set of distributions consistent with the conditional independence relationships defined by the undirected graph.
  - The set of distributions consistent with the factorization defined by potential functions on cliques of the graph.
- The Hammersley-Clifford theorem states that these two sets of distributions are the same.

The Markov blanket in an undirected graph



- This is simpler than in a directed graph because we do not have to worry about explaining away.
- The Markov blanket of a node is simply all of the directly connected nodes.

Image denoising with an MRF



- The true value of a pixel is x and the measured noisy value is y.
- We can define an energy function on pairs of nodes.

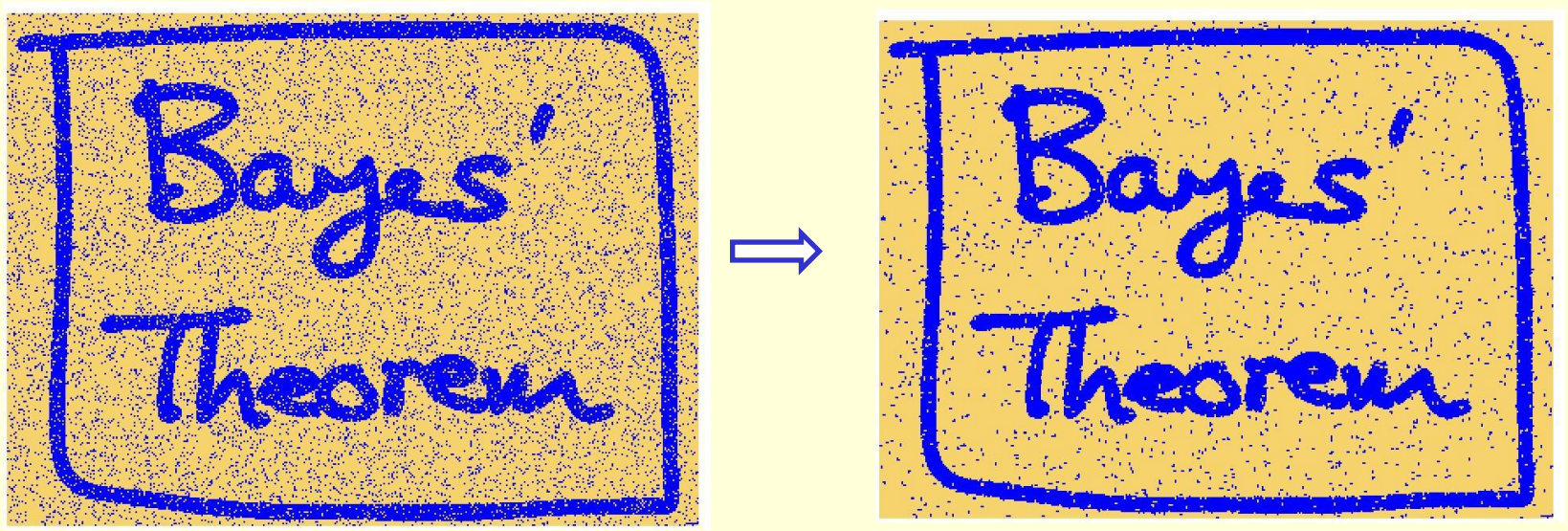
$$E(\mathbf{x},\mathbf{y}) = h\sum_i x_i - \beta\sum_{i<j} x_i x_j + \eta\sum_i (x_i - y_i)^2$$

$$(x_i - y_i)^2 = x_i^2 - 2x_i y_i + y_i^2$$ so we could use $-x_i y_i$

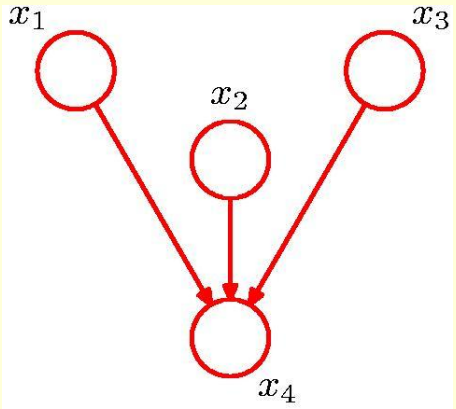
bias irrelevant

A simple, greedy MAP inference procedure

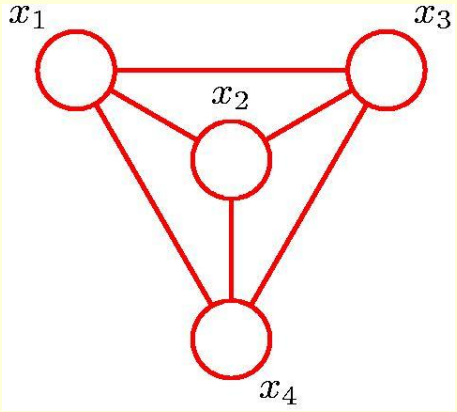
- Iterated conditional modes: Visit the unobserved nodes sequentially and set each x to whichever of its two values has the lowest energy.
  - This only requires us to look at the Markov blanket, i.e. The connected nodes.
- It would be better to flip in order of confidence.



Directed graphs can be more precise about independencies than undirected ones

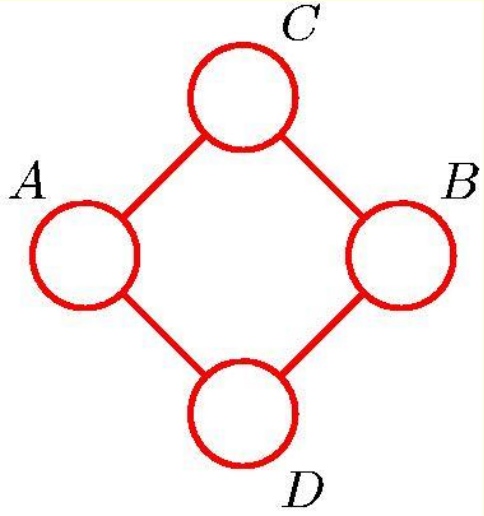


- All the parents of x_4 can interact to determine the distribution over x_4.
- The directed graph represents independencies that the undirected graph cannot model.



- To represent the high-order interaction in the directed graph, the undirected graph needs a fourth-order clique.
- So this graph cannot represent any independencies.

Undirected graphs can be more precise about independencies than directed ones



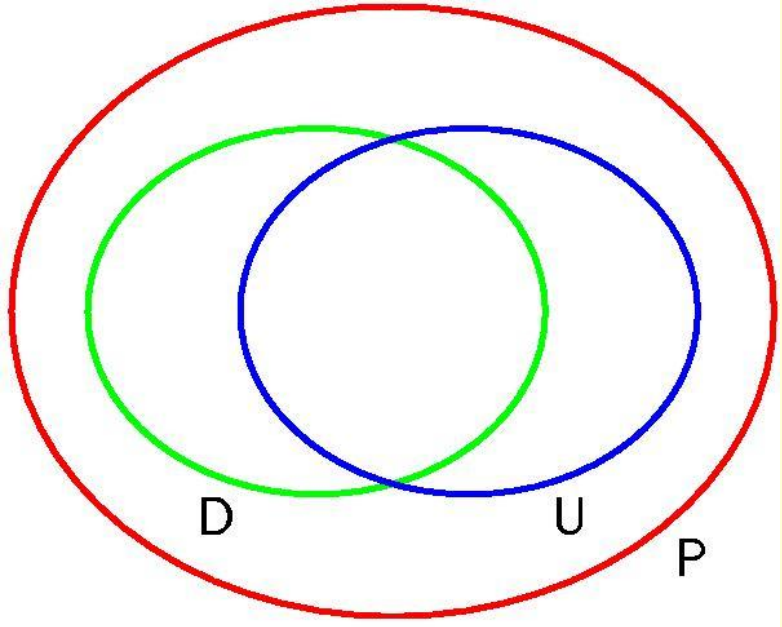
$$A \not\perp\!\!\!\perp B \mid 0$$

$$C \perp\!\!\!\perp D \mid A \cup B$$

$$A \perp\!\!\!\perp B \mid C \cup D$$

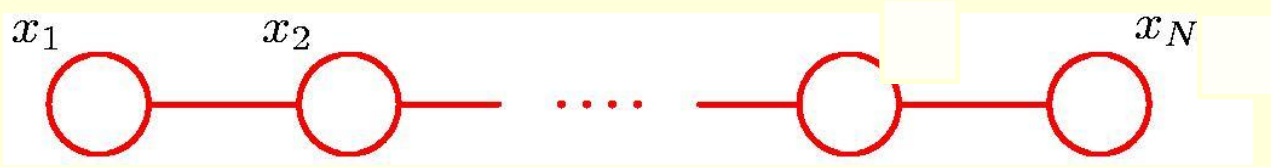
- This graph exhibits three independence properties that cannot all be exhibited by any directed graph.

The distributions for which directed and undirected graphs can give perfect maps



- A graph is a perfect map of a distribution if its conditional independencies are exactly the same asa those in the distribution.

Inference in an undirected chain



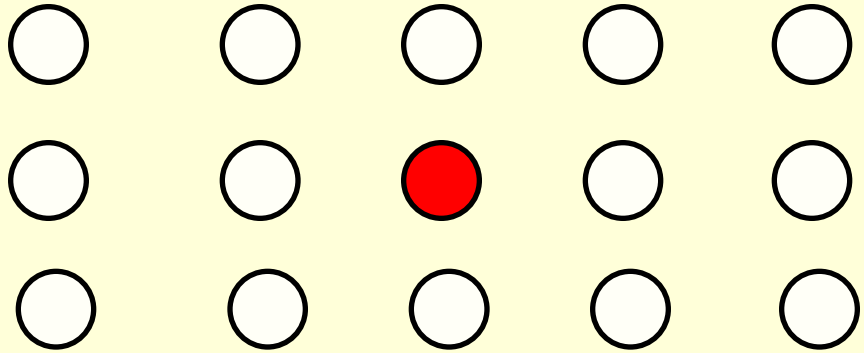
$$p(\mathbf{x}) = \frac{1}{Z}\varphi_{1,2}(x_1, x_2)\varphi_{2,3}(x_2, x_3)\ldots\ldots\varphi_{N-1,N}(x_{N-1}, x_N)$$

- Assume each node is a K-state discrete variable and each potential is a K x K table.
- Consider trying to compute the marginal distribution over the n'th node by summing over all values of all other nodes.

$$p(x_n) = \sum_{x_1}\ldots\ldots\sum_{x_{n-1}}\sum_{x_{n+1}}\ldots\ldots\sum_{x_N} p(\mathbf{x})$$

$$= \text{left branch term} \times \text{right branch term}$$

A picture of the computation



$$p(x_3) = \sum_{x_1} \sum_{x_2} \sum_{x_4} \sum_{x_5} p(x_1, x_2, x_3, x_4, x_5)$$

The recursive expressions for the left-branch and right-branch messages

$$\mu_\alpha(x_n) = \sum_{x_{n-1}} \varphi_{n-1,n}(x_{n-1}, x_n) \ldots \left[\sum_{x_2} \varphi_{2,3}(x_2, x_3) \left[\sum \varphi_{1,2}(x_1, x_2) \right] \right]$$

$$= \sum_{x_{n-1}} \varphi_{n-1,n}(x_{n-1}, x_n)\ \mu_\alpha(x_{n-1})$$

$$\mu_\beta(x_n) = \sum_{x_{n+1}} \varphi_{n,n+1}(x_n, x_{n+1}) \ldots \left[\sum_{x_N} \varphi_{N-1,N}(x_{N-1}, x_N) \right]$$

$$= \sum_{x_{n+1}} \varphi_{n,n+1}(x_n, x_{n+1})\ \mu_\beta(x_{n+1})$$

Computing more than one marginal distribution

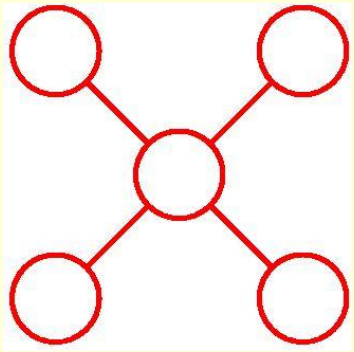
- First do a complete forward pass and a complete backward pass.
- Then the marginal for node n is:

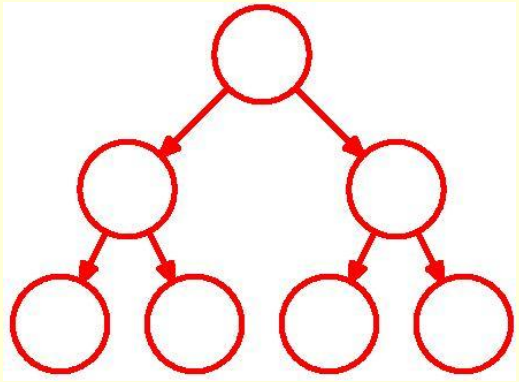
$$p(x_n) = \frac{1}{Z} \mu_\alpha(x_n)\, \mu_\beta(x_n)$$

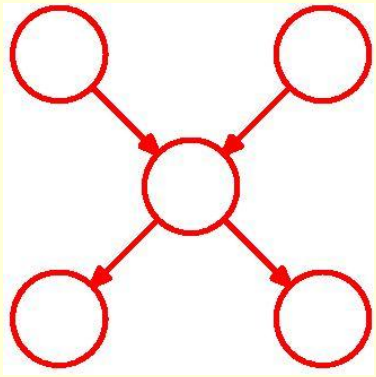
- The marginal for an adjacent pair of nodes is:

$$p(x_{n-1}, x_n) = \frac{1}{Z} \mu_\alpha(x_{n-1})\ \ \varphi_{n-1,n}(x_{n-1}, x_n)\ \ \mu_\beta(x_n)$$

Generalizing the inference procedure to trees

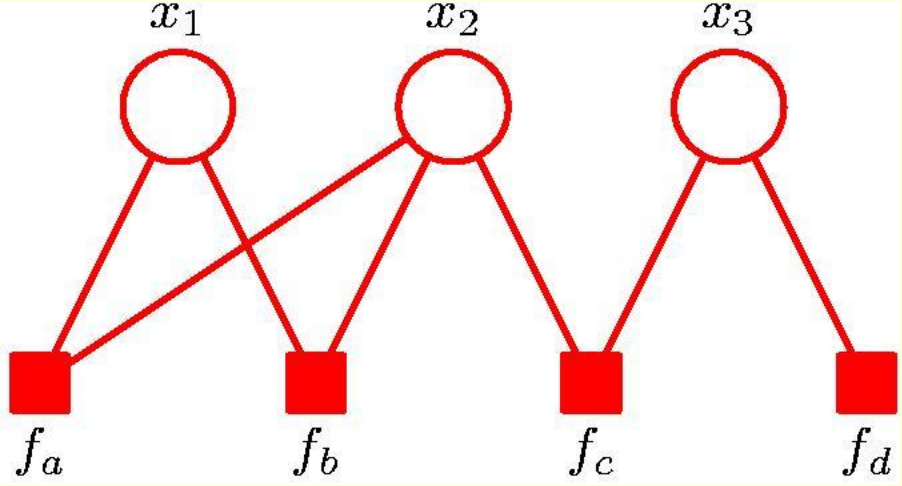






- The message passing procedure generalizes easily to any graph which is “singly connected”.
  - This includes trees and polytrees.
- Each node needs to send out along each link the product of the messages it receives on its other links.

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

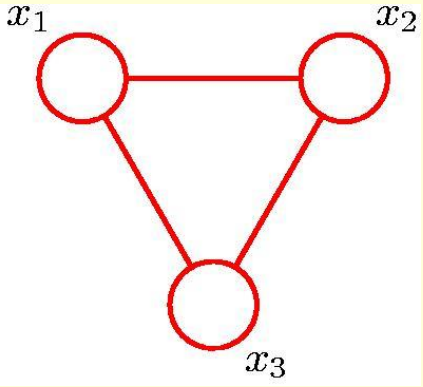


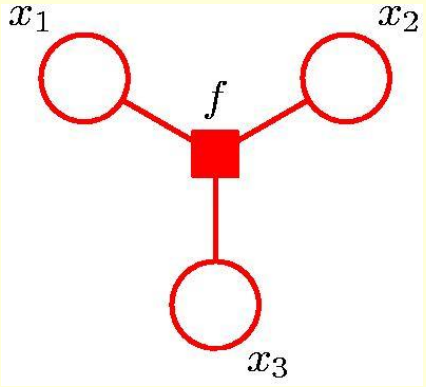
$$p(\mathbf{x}) = \prod_s f_s(\mathbf{x}_s)$$

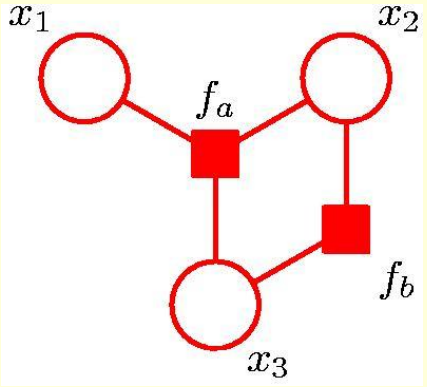
If the potentials are not normalized we need an extra factor of 1/Z.

- 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

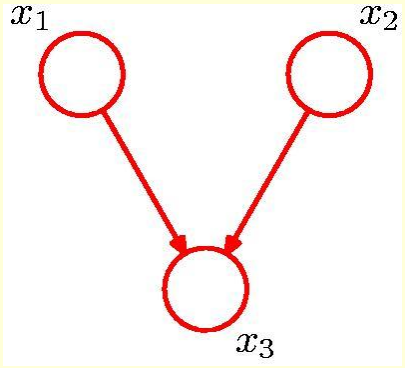


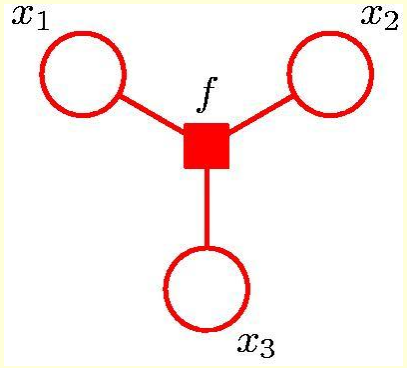


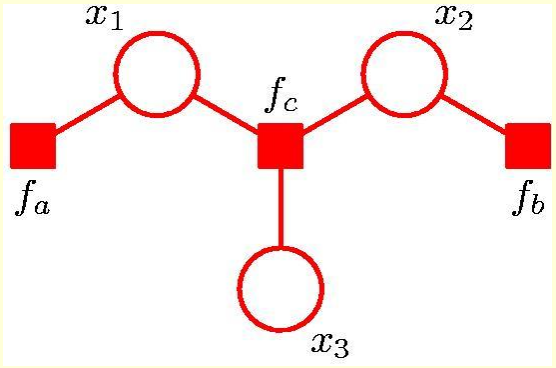


- The third-order factor is much more visually apparent than the clique of size 3.
- It easy to divide a factor into the product of several simpler factors.
  - This allows additional factorization to be represented.

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.

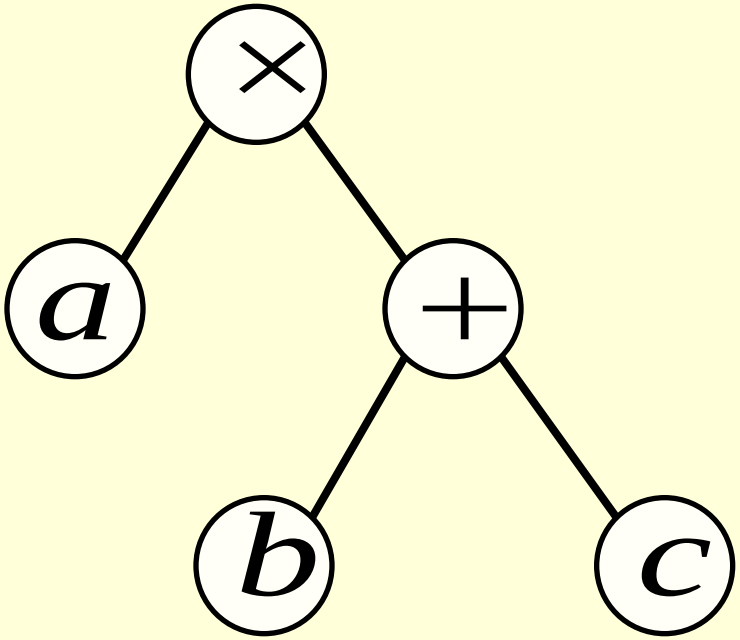
Computing a marginal in a factor graph with nodes that have discrete values

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

- 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

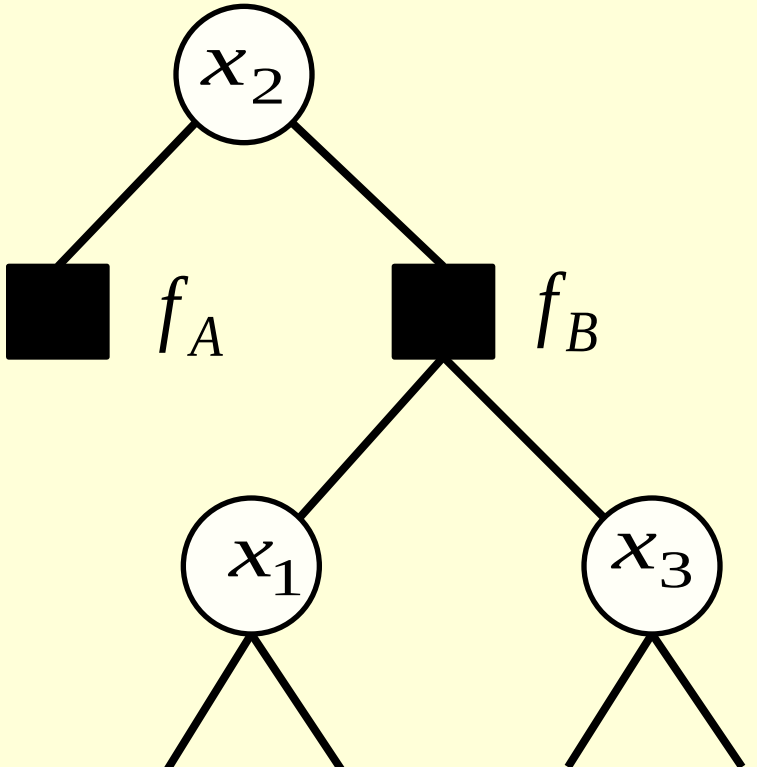
$$ab + ac = a(b + c)$$

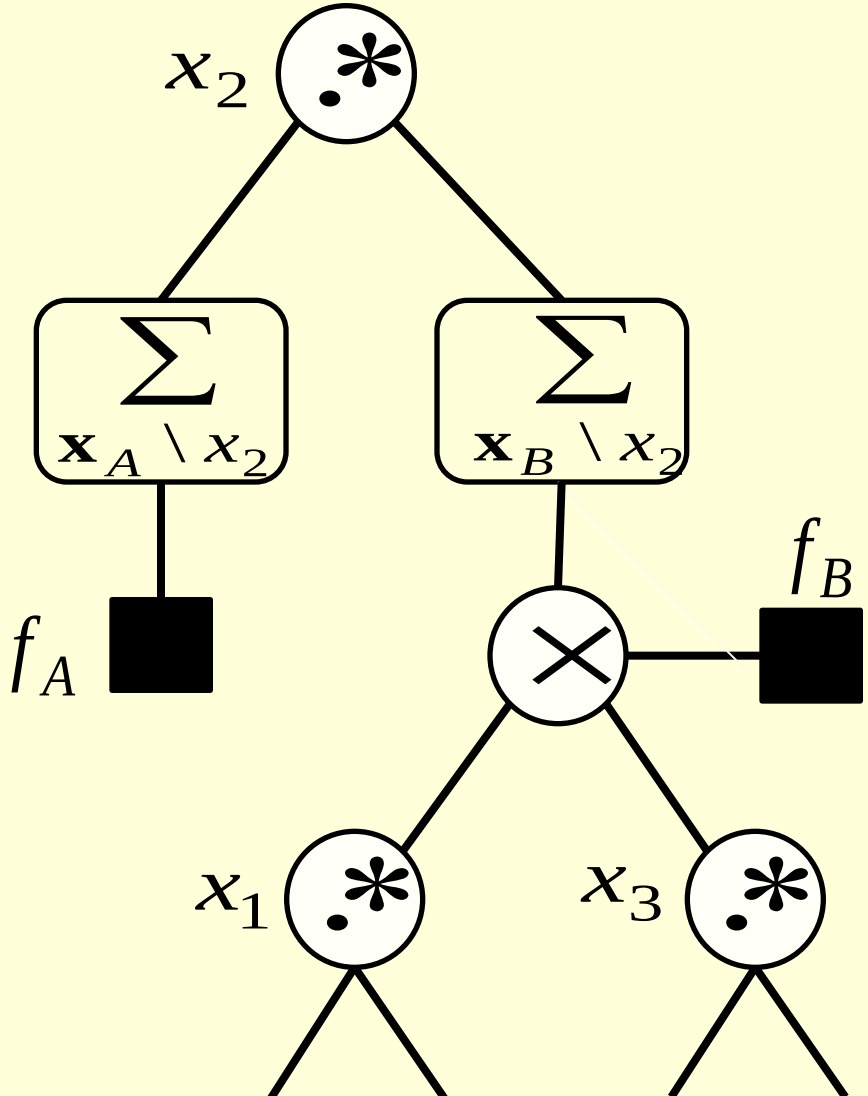


- We can compute the values of arithmetic expressions in a tree.
- We can do the same thing using probability distributions instead of scalar values.
  - The product operation gets replaced by a pointwise product.
  - The sum operation get replaced by something more complicated.

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

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

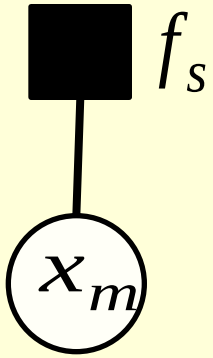
sum over all combinations of values of variables below

variables below

The messages at the leaf nodes

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

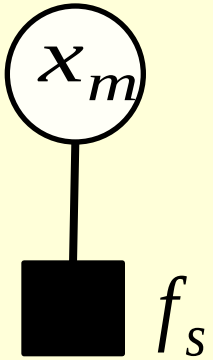
$$\mu_{x_m \to f_s}(x_m) = 1$$



leaf

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

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



leaf

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!