

Graphical Models

Joint Distribution

	<i>toothache</i>		$\neg$ <i>toothache</i>	
	<i>catch</i>	$\neg$ <i>catch</i>	<i>catch</i>	$\neg$ <i>catch</i>
<i>cavity</i>	.108	.012	.072	.008
$\neg$ <i>cavity</i>	.016	.064	.144	.576

- Consider model with 3 boolean random variables: *cavity*, *catch*, *toothache*
- Can answer query such as

$$p(\neg \text{cavity} | \text{toothache})$$

Joint Distribution

	<i>toothache</i>		$\neg$ <i>toothache</i>	
	<i>catch</i>	$\neg$ <i>catch</i>	<i>catch</i>	$\neg$ <i>catch</i>
<i>cavity</i>	.108	.012	.072	.008
$\neg$ <i>cavity</i>	.016	.064	.144	.576

- Consider model with 3 boolean random variables: *cavity*, *catch*, *toothache*
- Can answer query such as

$$p(\neg \text{cavity} | \text{toothache})$$

Joint Distribution

	<i>toothache</i>		$\neg$ <i>toothache</i>	
	<i>catch</i>	$\neg$ <i>catch</i>	<i>catch</i>	$\neg$ <i>catch</i>
<i>cavity</i>	.108	.012	.072	.008
$\neg$ <i>cavity</i>	.016	.064	.144	.576

- Consider model with 3 boolean random variables: *cavity*, *catch*, *toothache*
- Can answer query such as

$$p(\neg \text{cavity} | \text{toothache}) = \frac{p(\neg \text{cavity}, \text{toothache})}{p(\text{toothache})}$$

$$p(\neg \text{cavity} | \text{toothache}) = \frac{0.016 + 0.064}{0.108 + 0.012 + 0.016 + 0.064} = 0.4$$

Joint Distribution

	<i>toothache</i>		$\neg$ <i>toothache</i>	
	<i>catch</i>	$\neg$ <i>catch</i>	<i>catch</i>	$\neg$ <i>catch</i>
<i>cavity</i>	.108	.012	.072	.008
$\neg$ <i>cavity</i>	.016	.064	.144	.576

- Consider model with 3 boolean random variables: *cavity*, *catch*, *toothache*
- Can answer query such as

$$p(\neg \text{cavity} | \text{toothache}) = \frac{p(\neg \text{cavity}, \text{toothache})}{p(\text{toothache})}$$

$$p(\neg \text{cavity} | \text{toothache}) = \frac{0.016 + 0.064}{0.108 + 0.012 + 0.016 + 0.064} = 0.4$$

Joint Distribution

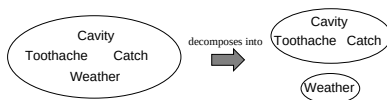
- In general, to answer a query on random variables $\mathbf{Q} = Q_1, \dots, Q_N$ given evidence $\mathbf{E} = \mathbf{e}$, $\mathbf{E} = E_1, \dots, E_M$, $\mathbf{e} = e_1, \dots, e_M$:

$$\begin{aligned} p(\mathbf{Q} | \mathbf{E} = \mathbf{e}) &= \frac{p(\mathbf{Q}, \mathbf{E} = \mathbf{e})}{p(\mathbf{E} = \mathbf{e})} \\ &= \frac{\sum_{\mathbf{h}} p(\mathbf{Q}, \mathbf{E} = \mathbf{e}, \mathbf{H} = \mathbf{h})}{\sum_{\mathbf{q}, \mathbf{h}} p(\mathbf{Q} = \mathbf{q}, \mathbf{E} = \mathbf{e}, \mathbf{H} = \mathbf{h})} \end{aligned}$$

Problems

- The joint distribution is large
 - e. g. with K boolean random variables, 2^K entries
- Inference is slow, previous summations take $O(2^K)$ time
- Learning is difficult, data for 2^K parameters
- Analogous problems for continuous random variables

Reminder - Independence



- A and B are **independent** iff
 $p(A|B) = p(A)$ or $p(B|A) = p(B)$ or $p(A, B) = p(A)p(B)$
- $p(\textit{Toothache}, \textit{Catch}, \textit{Cavity}, \textit{Weather}) =$
 $p(\textit{Toothache}, \textit{Catch}, \textit{Cavity})p(\textit{Weather})$
 - 32 entries reduced to 12 (*Weather* takes one of 4 values)
- Absolute independence powerful but rare
- Dentistry is a large field with hundreds of variables, none of which are independent. What to do?

Reminder - Conditional Independence

- $p(\textit{Toothache}, \textit{Cavity}, \textit{Catch})$ has $2^3 - 1 = 7$ independent entries
- If I have a cavity, the probability that the probe catches in it doesn't depend on whether I have a toothache:
(1) $P(\textit{catch}|\textit{toothache}, \textit{cavity}) = P(\textit{catch}|\textit{cavity})$
- The same independence holds if I haven't got a cavity:
(2) $P(\textit{catch}|\textit{toothache}, \neg\textit{cavity}) = P(\textit{catch}|\neg\textit{cavity})$
- *Catch* is **conditionally independent** of *Toothache* given *Cavity*: $p(\textit{Catch}|\textit{Toothache}, \textit{Cavity}) = p(\textit{Catch}|\textit{Cavity})$
- Equivalent statements:
 - $p(\textit{Toothache}|\textit{Catch}, \textit{Cavity}) = p(\textit{Toothache}|\textit{Cavity})$
 - $p(\textit{Toothache}, \textit{Catch}|\textit{Cavity}) = p(\textit{Toothache}|\textit{Cavity})p(\textit{Catch}|\textit{Cavity})$
 - $\textit{Toothache} \perp\!\!\!\perp \textit{Catch}|\textit{Cavity}$

Conditional Independence contd.

- Write out full joint distribution using chain rule:

$$p(\textit{Toothache}, \textit{Catch}, \textit{Cavity})$$

$$= p(\textit{Toothache} | \textit{Catch}, \textit{Cavity}) p(\textit{Catch}, \textit{Cavity})$$

$$= p(\textit{Toothache} | \textit{Catch}, \textit{Cavity}) p(\textit{Catch} | \textit{Cavity}) p(\textit{Cavity})$$

$$= p(\textit{Toothache} | \textit{Cavity}) p(\textit{Catch} | \textit{Cavity}) p(\textit{Cavity})$$

2 + 2 + 1 = 5 independent numbers

- In many cases, the use of conditional independence greatly reduces the size of the representation of the joint distribution

Graphical Models

- Graphical Models provide a visual depiction of probabilistic model
- Conditional independence assumptions can be seen in graph
- Inference and learning algorithms can be expressed in terms of graph operations
- We will look at 3 types of graph (can be combined)
 - Directed graphs: [Bayesian networks](#)
 - Undirected graphs: [Markov Random Fields](#)
 - [Factor graphs](#)

Outline

Probabilistic Models

Bayesian Networks

Markov Random Fields

Inference

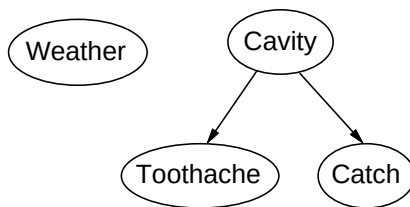
Bayesian Networks

- A simple, graphical notation for conditional independence assertions and hence for compact specification of full joint distributions
- Syntax:
 - a set of nodes, one per variable
 - a directed, acyclic graph (link $\approx$ “directly influences”)
 - a conditional distribution for each node given its parents:

$$p(X_i | pa(X_i))$$

- In the simplest case, conditional distribution represented as a **conditional probability table** (CPT) giving the distribution over X_i for each combination of parent values

Example

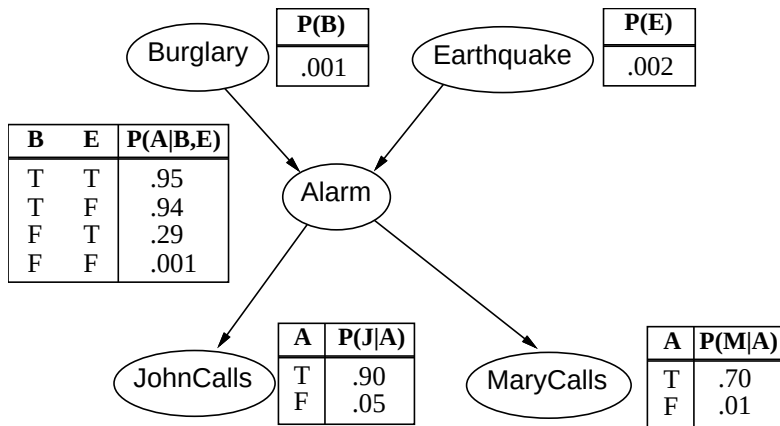


- Topology of network encodes conditional independence assertions:
 - *Weather* is independent of the other variables
 - *Toothache* and *Catch* are conditionally independent given *Cavity*

Example

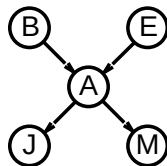
- I'm at work, neighbor John calls to say my alarm is ringing, but neighbor Mary doesn't call. Sometimes it's set off by minor earthquakes. Is there a burglar?
- Variables: *Burglar*, *Earthquake*, *Alarm*, *JohnCalls*, *MaryCalls*
- Network topology reflects causal knowledge:
 - A burglar can set the alarm off
 - An earthquake can set the alarm off
 - The alarm can cause Mary to call
 - The alarm can cause John to call
- (Causal models and conditional independence seem hardwired for humans!)

Example contd.



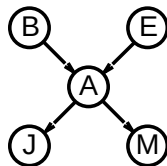
Compactness

- A CPT for Boolean X_i with k Boolean parents has 2^k rows for the combinations of parent values
- Each row requires one number p for $X_i = \text{true}$ (the number for $X_i = \text{false}$ is just $1 - p$)
- If each variable has no more than k parents, the complete network requires $O(n \cdot 2^k)$ numbers
- i.e., grows linearly with n , vs. $O(2^n)$ for the full joint distribution
- For burglary net, ?? numbers



Compactness

- A CPT for Boolean X_i with k Boolean parents has 2^k rows for the combinations of parent values
- Each row requires one number p for $X_i = \text{true}$ (the number for $X_i = \text{false}$ is just $1 - p$)
- If each variable has no more than k parents, the complete network requires $O(n \cdot 2^k)$ numbers
- i.e., grows linearly with n , vs. $O(2^n)$ for the full joint distribution
- For burglary net, ?? numbers
 - $1 + 1 + 4 + 2 + 2 = 10$ numbers (vs. $2^5 - 1 = 31$)

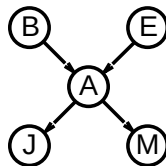


Global Semantics

- **Global semantics** defines the full joint distribution as the product of the local conditional distributions:

$$P(x_1, \dots, x_n) = \prod_{i=1}^n P(x_i | pa(X_i))$$

e.g., $P(j \wedge m \wedge a \wedge \neg b \wedge \neg e) =$



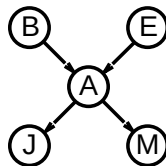
Global Semantics

- **Global semantics** defines the full joint distribution as the product of the local conditional distributions:

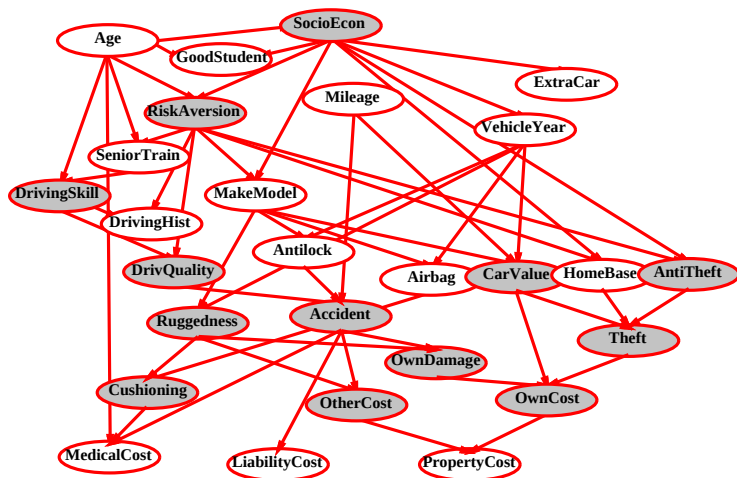
$$P(x_1, \dots, x_n) = \prod_{i=1}^n P(x_i | pa(X_i))$$

e.g., $P(j \wedge m \wedge a \wedge \neg b \wedge \neg e) =$

$$\begin{aligned} & P(j|a)P(m|a)P(a|\neg b, \neg e)P(\neg b)P(\neg e) \\ = & 0.9 \times 0.7 \times 0.001 \times 0.999 \times 0.998 \\ \approx & 0.00063 \end{aligned}$$



Example - Car Insurance



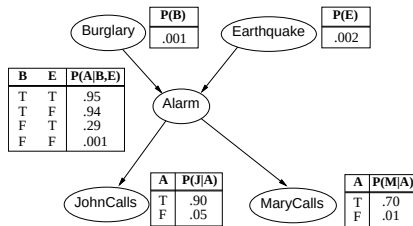
<http://aispace.org/bayes>

Specifying Distributions - Discrete Variables

- Earlier we saw the use of **conditional probability tables** (CPT) for specifying a distribution over discrete random variables with discrete-valued parents
- For a variable with no parents, with K possible states:

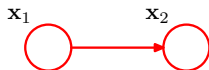
$$p(\mathbf{x}|\boldsymbol{\mu}) = \prod_{k=1}^K \mu_k^{x_k}$$

- e.g. $p(B) = 0.001^{B_1} 0.999^{B_2}$,
1-of- K representation



Specifying Distributions - Discrete Variables cont.

- With two variables x_1, x_2 can have two cases



- Dependent

$$\begin{aligned}
 p(x_1, x_2 | \mu) &= p(x_1 | \mu) p(x_2 | x_1, \mu) \\
 &= \left(\prod_{k=1}^K \mu_{k1}^{x_{1k}} \right) \left(\prod_{k=1}^K \prod_{j=1}^K \mu_{kj2}^{x_{1k} x_{2j}} \right)
 \end{aligned}$$

- $K^2 - 1$ free parameters in μ



- Independent

$$\begin{aligned}
 p(x_1, x_2 | \mu) &= p(x_1 | \mu) p(x_2 | \mu) \\
 &= \left(\prod_{k=1}^K \mu_{k1}^{x_{1k}} \right) \left(\prod_{k=1}^K \mu_{k2}^{x_{2k}} \right)
 \end{aligned}$$

- $2(K - 1)$ free parameters in μ

Chains of Nodes

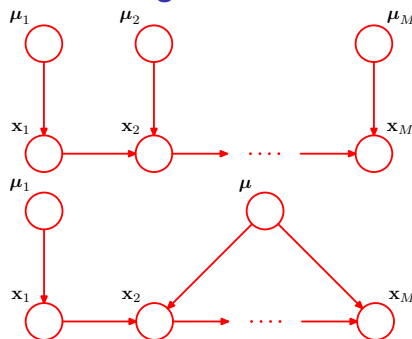


- With M nodes, could form a chain as shown above
- Number of parameters is:

$$\underbrace{(K - 1)}_{x_1} + (M - 1) \underbrace{K(K - 1)}_{others}$$

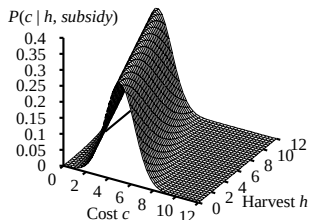
- Compare to:
 - $K^M - 1$ for fully connected graph
 - $M(K - 1)$ for graph with no edges (all independent)

Sharing Parameters



- Another way to reduce number of parameters is **sharing** parameters (a. k. a. **tying** of parameters)
- Lower graph reuses same μ for nodes 2- M
 - μ is a random variable in this network, could also be deterministic
- $(K - 1) + K(K - 1)$ parameters

Specifying Distributions - Continuous Variables



- One common type of conditional distribution for continuous variables is the **linear-Gaussian**

$$p(x_i | pa_i) = \mathcal{N} \left(x_i; \sum_{j \in pa_i} w_{ij} x_j + b_i, v_i \right)$$

- e.g. With one parent *Harvest*:

$$p(c|h) = \mathcal{N}(c; -0.5h + 5, 1)$$

- For harvest h , mean cost is $-0.5h + 5$, variance is 1

Linear Gaussian

- Interesting fact: if all nodes in a Bayesian Network are linear Gaussian, joint distribution is a multivariate Gaussian.
- Converse is true as well, see Ch.2.3.

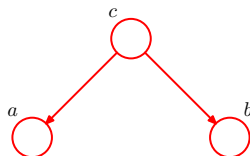
$$p(x_i|pa_i) = \mathcal{N} \left(x_i; \sum_{j \in pa_i} w_{ij}x_j + b_i, v_i \right)$$
$$p(x_1, \dots, x_N) = \prod_{i=1}^N \mathcal{N} \left(x_i; \sum_{j \in pa_i} w_{ij}x_j + b_i, v_i \right)$$

- Each factor looks like $\exp(-(x_i - (\mathbf{w}_i^T \mathbf{x}_{pa_i}))^2)$, this product will be another quadratic form of the components of $\mathbf{x}$.
- With no links in graph, end up with diagonal covariance matrix
- With fully connected graph, end up with full covariance matrix

Conditional Independence in Bayesian Networks

- Recall again that a and b are conditionally independent given c ($a \perp\!\!\!\perp b|c$) if
 - $p(a|b, c) = p(a|c)$ or equivalently
 - $p(a, b|c) = p(a|c)p(b|c)$
- Before we stated that links in a graph are $\approx$ “direct influences”
- We now develop a correct notion of links, in terms of the conditional independences they represent
 - This will be useful for general-purpose inference methods
 - It provides a fast solution to the *relevance problem*:
determine whether X is relevant to Y given knowledge of Z .

A Tale of Three Graphs - Part 1

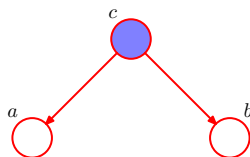


- The graph above means

$$\begin{aligned} p(a, b, c) &= p(a|c)p(b|c)p(c) \\ p(a, b) &= \sum_c p(a|c)p(b|c)p(c) \\ &\neq p(a)p(b) \text{ in general} \end{aligned}$$

- So a and b not independent

A Tale of Three Graphs - Part 1

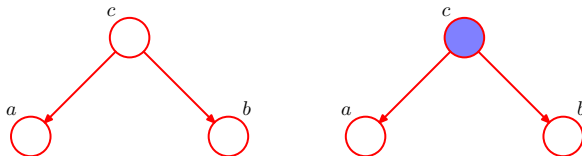


- However, conditioned on c

$$p(a, b|c) = \frac{p(a, b, c)}{p(c)} = \frac{p(a|c)p(b|c)p(c)}{p(c)} = p(a|c)p(b|c)$$

- So $a \perp\!\!\!\perp b|c$

A Tale of Three Graphs - Part 1



- Note the **path** from a to b in the graph
 - When c is not observed, path is open, a and b not independent
 - When c is observed, path is blocked, a and b independent
- In this case c is **tail-to-tail** with respect to this path

A Tale of Three Graphs - Part 2

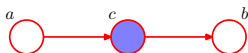


- The graph above means

$$p(a, b, c) = p(a)p(b|c)p(c|a)$$

- Again a and b not independent

A Tale of Three Graphs - Part 2



- However, conditioned on c

$$\begin{aligned} p(a, b|c) &= \frac{p(a, b, c)}{p(c)} = \frac{p(a)p(b|c)}{p(c)} p(c|a) \\ &= \frac{p(a)p(b|c)}{p(c)} \underbrace{\frac{p(a|c)p(c)}{p(a)}}_{\text{Bayes' Rule}} \\ &= p(a|c)p(b|c) \end{aligned}$$

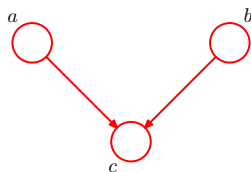
- So $a \perp\!\!\!\perp b|c$

A Tale of Three Graphs - Part 2



- As before, the **path** from a to b in the graph
 - When c is not observed, path is open, a and b not independent
 - When c is observed, path is blocked, a and b independent
- In this case c is **head-to-tail** with respect to this path

A Tale of Three Graphs - Part 3

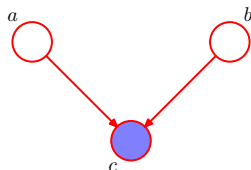


- The graph above means

$$\begin{aligned} p(a, b, c) &= p(a)p(b)p(c|a, b) \\ p(a, b) &= \sum_c p(a)p(b)p(c|a, b) \\ &= p(a)p(b) \end{aligned}$$

- This time a and b are independent

A Tale of Three Graphs - Part 3

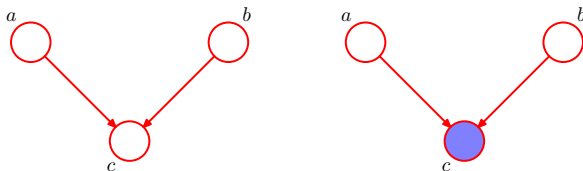


- However, conditioned on c

$$\begin{aligned} p(a, b|c) &= \frac{p(a, b, c)}{p(c)} = \frac{p(a)p(b)p(c|a, b)}{p(c)} \\ &\neq p(a|c)p(b|c) \text{ in general} \end{aligned}$$

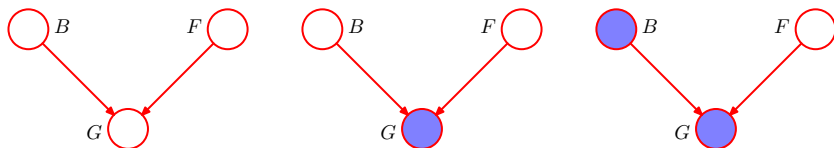
- So a is dependent on b given c

A Tale of Three Graphs - Part 3



- The behaviour here is different
 - When c is not observed, path is blocked, a and b independent
 - When c is observed, path is unblocked, a and b not independent
- In this case c is **head-to-head** with respect to this path
- Situation is in fact more complex, path is unblocked if any **descendent** of c is observed

Part 3 - Intuition

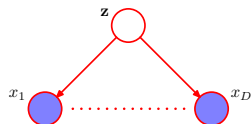


- Binary random variables B (battery charged), F (fuel tank full), G (fuel gauge reads full)
- B and F independent
- But if we observe $G = 0$ (false) things change
 - e.g. $p(F = 0|G = 0, B = 0)$ could be less than $p(F = 0|G = 0)$, as $B = 0$ **explains away** the fact that the gauge reads empty
 - Recall that $p(F|G, B) = p(F|G)$ is another $F \perp\!\!\!\perp B|G$

D-separation

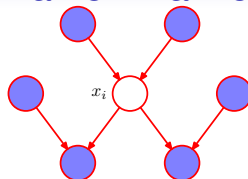
- A general statement of conditional independence
- For sets of nodes A , B , C , check all paths from A to B in graph
- If all paths are **blocked**, then $A \perp\!\!\!\perp B | C$
- Path is blocked if:
 - Arrows meet **head-to-tail** or **tail-to-tail** at a node in C
 - Arrows meet **head-to-head** at a node, and neither node nor any descendent is in C

Naive Bayes



- Commonly used **naive Bayes** classification model
- Class label z , features $x_1, \dots, x_D$
- Model assumes features independent given class label
 - **Tail-to-tail** at z , blocks path between features

Markov Blanket



- What is the minimal set of nodes which makes a node x_i conditionally independent from the rest of the graph?
 - x_i 's parents, children, and children's parents (co-parents)
- Define this set MB , and consider:

$$\begin{aligned}
 p(x_i | x_{\{j \neq i\}}) &= \frac{p(x_1, \dots, x_D)}{\int p(x_1, \dots, x_D) dx_i} \\
 &= \frac{\prod_k p(x_k | pa_k)}{\int \prod_k p(x_k | pa_k) dx_i}
 \end{aligned}$$

- All factors other than those for which x_i is x_k or in pa_k cancel

Learning Parameters

- When all random variables are observed in training data, relatively straight-forward
 - Distribution factors, all factors observed
 - e.g. Maximum likelihood used to set parameters of each distribution $p(x_i|pa_i)$ separately
- When some random variables not observed, it's tricky
 - This is a common case
 - Expectation-maximization (later) is a method for this

Outline

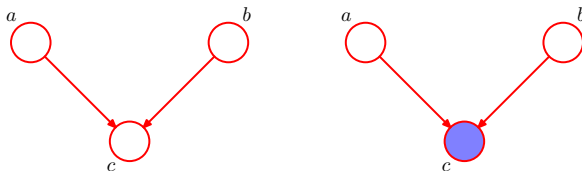
Probabilistic Models

Bayesian Networks

Markov Random Fields

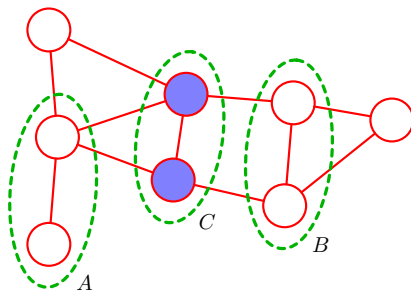
Inference

Conditional Independence in Graphs



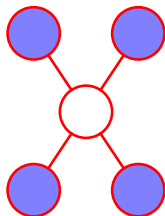
- Recall that for Bayesian Networks, conditional independence was a bit complicated
 - **d-separation** with head-to-head links
- We would like to construct a graphical representation such that conditional independence is straight-forward path checking

Markov Random Fields



- **Markov random fields** (MRFs) contain one node per variable
- Undirected graph over these nodes
- Conditional independence will be given by simple separation, blockage by observing a node on a path
 - e.g. in above graph, $A \perp\!\!\!\perp B | C$

Markov Blanket Markov



- With this simple check for conditional independence, **Markov blanket** is also simple
 - Recall Markov blanket MB of x_i is set of nodes such that x_i conditionally independent from rest of graph given MB
- Markov blanket is neighbours

MRF Factorization

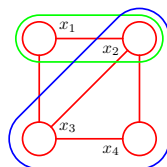
- Remember that graphical models define a factorization of the joint distribution
- What should be the factorization so that we end up with the simple conditional independence check?
- For x_i and x_j not connected by an edge in graph:

$$x_i \perp\!\!\!\perp x_j \mid \mathbf{x}_{\setminus \{i,j\}}$$

- So there should not be any factor $\psi(x_i, x_j)$ in the factorized form of the joint

Cliques

- A **clique** in a graph is a subset of nodes such that there is a link between every pair of nodes in the subset
- A **maximal clique** is a clique for which one cannot add another node and have the set remain a clique



MRF Joint Distribution

- Note that nodes in a clique cannot be made conditionally independent from each other
 - So defining factors $\psi(\cdot)$ on nodes in a clique is “safe”
- The joint distribution for a Markov random field is:

$$p(x_1, \dots, x_K) = \frac{1}{Z} \prod_C \psi_C(\mathbf{x}_C)$$

where $\mathbf{x}_C$ is the set of nodes in clique C , and the product runs over all maximal cliques

- Each $\psi_C(\mathbf{x}_C) \geq 0$
- Z is a normalization constant

MRF Joint - Terminology

- The joint distribution for a Markov random field is:

$$p(x_1, \dots, x_K) = \frac{1}{Z} \prod_C \psi_C(\mathbf{x}_C)$$

- Each $\psi_C(\mathbf{x}_C) \geq 0$ is called a **potential function**
- Z , the normalization constant, is called the **partition function**:

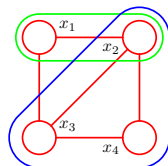
$$Z = \sum_{\mathbf{x}} \prod_C \psi_C(\mathbf{x}_C)$$

- Z is very costly to compute, since it is a sum/integral over all possible states for all variables in $\mathbf{x}$
- Don't always need to evaluate it though, will cancel for computing conditional probabilities

MRF Joint Distribution Example

- The joint distribution for a Markov random field is:

$$\begin{aligned} p(x_1, \dots, x_4) &= \frac{1}{Z} \prod_C \psi_C(\mathbf{x}_C) \\ &= \frac{1}{Z} \psi_{123}(x_1, x_2, x_3) \psi_{234}(x_2, x_3, x_4) \end{aligned}$$



- Note that maximal cliques subsume smaller ones: $\psi_{123}(x_1, x_2, x_3)$ could include $\psi_{12}(x_1, x_2)$, though sometimes smaller cliques are explicitly used for clarity

Hammersley-Clifford

- The definition of the joint:

$$p(x_1, \dots, x_K) = \frac{1}{Z} \prod_C \psi_C(\mathbf{x}_C)$$

- Note that we started with particular conditional independences
- We then formulated the factorization based on clique potentials
 - This formulation resulted in the right conditional independences
- The converse is true as well, any strictly positive distribution with the conditional independences given by the undirected graph **can** be represented using a product of clique potentials
- This is the **Hammersley-Clifford** theorem

Energy Functions

- Often use exponential, which is non-negative, to define potential functions:

$$\psi_C(\mathbf{x}_C) = \exp\{-E_C(\mathbf{x}_C)\}$$

- Minus sign – by convention
- $E_C(\mathbf{x}_C)$ is called an **energy function**
 - From physics, low energy = high probability
- This exponential representation is known as the **Boltzmann distribution**

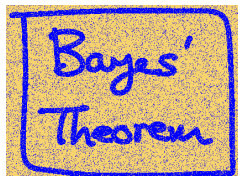
Energy Functions - Intuition

- Joint distribution nicely rearranges as

$$\begin{aligned} p(x_1, \dots, x_K) &= \frac{1}{Z} \prod_C \psi_C(\mathbf{x}_C) \\ &= \frac{1}{Z} \exp\left\{-\sum_C E_C(\mathbf{x}_C)\right\} \end{aligned}$$

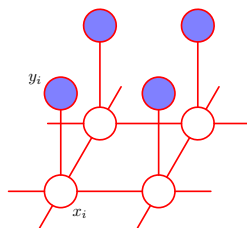
- Intuition about potential functions: ψ_C are describing good (low energy) sets of states for adjacent nodes
- An example of this is next

Image Denoising



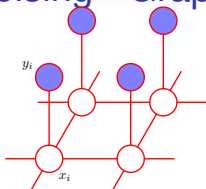
- Consider the problem of trying to correct (denoise) an image that has been corrupted
- Assume image is binary
- Observed (noisy) pixel values $y_i \in \{-1, +1\}$
- Unobserved true pixel values $x_i \in \{-1, +1\}$
- Another application: face sketch synthesis from photos
<http://people.csail.mit.edu/xgwang/sketch.html>.

Image Denoising - Graphical Model



- Cliques containing each true pixel value $x_i \in \{-1, +1\}$ and observed value $y_i \in \{-1, +1\}$
 - Observed pixel value is usually same as true pixel value
 - Energy function $-\eta x_i y_i$, $\eta > 0$, lower energy (better) if $x_i = y_i$
- Cliques containing adjacent true pixel values x_i, x_j
 - Nearby pixel values are usually the same
 - Energy function $-\beta x_i x_j$, $\beta > 0$, lower energy (better) if $x_i = x_j$

Image Denoising - Graphical Model



- Complete energy function:

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

- Joint distribution:

$$p(\mathbf{x}, \mathbf{y}) = \frac{1}{Z} \exp\{-E(\mathbf{x}, \mathbf{y})\}$$

- Or, as potential functions $\psi_n(x_i, x_j) = \exp(\beta x_i x_j)$,
 $\psi_p(x_i, y_i) = \exp(\eta x_i y_i)$:

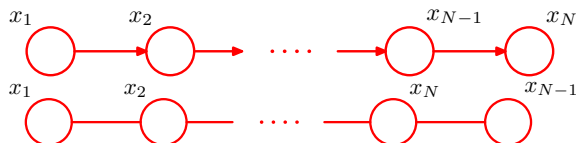
$$p(\mathbf{x}, \mathbf{y}) = \frac{1}{Z} \prod_{i,j} \psi_n(x_i, x_j) \prod_i \psi_p(x_i, y_i)$$

Image Denoising - Inference



- The denoising query is $\arg \max_x p(x|y)$
- Two approaches:
 - **Iterated conditional modes** (ICM): hill climbing in x , one variable x_i at a time
 - Simple to compute, Markov blanket is just observation plus neighbouring pixels
 - **Graph cuts**: formulate as max-flow/min-cut problem, exact inference (for this graph)

Converting Directed Graphs into Undirected Graphs



- Consider a simple directed chain graph:

$$p(\mathbf{x}) = p(x_1)p(x_2|x_1)p(x_3|x_2) \dots p(x_N|x_{N-1})$$

- Can convert to undirected graph

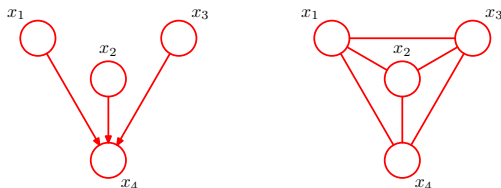
$$p(\mathbf{x}) = \frac{1}{Z} \psi_{1,2}(x_1, x_2) \psi_{2,3}(x_2, x_3) \dots \psi_{N-1,N}(x_{N-1}, x_N)$$

where $\psi_{1,2} = p(x_1)p(x_2|x_1)$, all other $\psi_{k-1,k} = p(x_k|x_{k-1})$,
 $Z = 1$

Converting Directed Graphs into Undirected Graphs

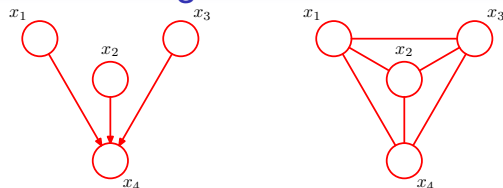
- The chain was straight-forward because for each conditional $p(x_i|pa_i)$, nodes $x_i \cup pa_i$ were contained in one clique
 - Hence we could define that clique potential to include that conditional
- For a general undirected graph we can force this to occur by “marrying” the parents
 - Add links between all parents in pa_i
 - This process known as **moralization**, creating a **moral graph**

Strong Morals



- Start with directed graph on left
- Add undirected edges between all parents of each node
- Remove directionality from original edges

Constructing Potential Functions



- Initialize all potential functions to be 1
- With moral graph, for each $p(x_i|pa_i)$, there is at least one clique which contains all of $x_i \cup pa_i$
 - Multiply $p(x_i|pa_i)$ into potential function for one of these cliques
- $Z = 1$ again since:

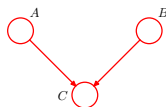
$$p(\mathbf{x}) = \prod_C \psi_C(\mathbf{x}_C) = \prod_i p(x_i|pa_i)$$

which is already normalized

Equivalence Between Graph Types

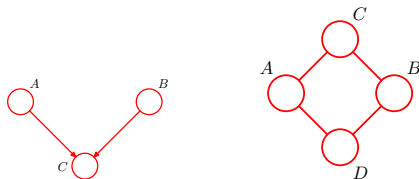
- Note that the moralized undirected graph loses some of the conditional independence statements of the directed graph
- Further, there are certain conditional independence assumptions which can be represented by directed graphs which **cannot** be represented by directed graphs, **and vice versa**
- Directed graph: $A \perp\!\!\!\perp B|\emptyset$, $A \perp\!\!\!\perp B|C$, cannot be represented using undirected graph
- Undirected graph: $A \perp\!\!\!\perp B|\emptyset$, $A \perp\!\!\!\perp B|C \cup D$, $C \perp\!\!\!\perp D|A \cup B$ cannot be represented using directed graph

Equivalence Between Graph Types



- Note that the moralized undirected graph loses some of the conditional independence statements of the directed graph
- Further, there are certain conditional independence assumptions which can be represented by directed graphs which **cannot** be represented by directed graphs, **and vice versa**
- Directed graph: $A \perp\!\!\!\perp B | \emptyset$, $A \perp\!\!\!\perp B | C$, cannot be represented using undirected graph
- Undirected graph: $A \perp\!\!\!\perp B | \emptyset$, $A \perp\!\!\!\perp B | C \cup D$, $C \perp\!\!\!\perp D | A \cup B$ cannot be represented using directed graph

Equivalence Between Graph Types



- Note that the moralized undirected graph loses some of the conditional independence statements of the directed graph
- Further, there are certain conditional independence assumptions which can be represented by directed graphs which **cannot** be represented by undirected graphs, **and vice versa**
- Directed graph: $A \perp\!\!\!\perp B | \emptyset$, $A \perp\!\!\!\perp B | C$, cannot be represented using undirected graph
- Undirected graph: $A \perp\!\!\!\perp B | \emptyset$, $A \perp\!\!\!\perp B | C \cup D$, $C \perp\!\!\!\perp D | A \cup B$ cannot be represented using directed graph

Inference

- **Inference** is the process of answering queries such as $p(x_n | \mathbf{x}_e = \mathbf{e})$
- We will focus on computing **marginal posterior distributions** over single variables x_n using

$$p(x_n | \mathbf{x}_e = \mathbf{e}) \propto p(x_n, \mathbf{x}_e = \mathbf{e})$$

- The proportionality constant can be obtained by enforcing $\sum_{x_n} p(x_n | \mathbf{x}_e = \mathbf{e}) = 1$

Inference on a Chain



- Consider a simple undirected chain
- For inference, we want to compute $p(x_n, \mathbf{x}_e = \mathbf{e})$
- First, we'll show how to compute $p(x_n)$
 - $p(x_n, \mathbf{x}_e = \mathbf{e})$ will be a simple modification of this

Inference on a Chain



- The naive method of computing the marginal $p(x_n)$ is to write down the factored form of the joint, and marginalize (sum out) all other variables:

$$\begin{aligned}
 p(x_n) &= \sum_{x_1} \cdots \sum_{x_{n-1}} \sum_{x_{n+1}} \cdots \sum_{x_N} p(\mathbf{x}) \\
 &= \sum_{x_1} \cdots \sum_{x_{n-1}} \sum_{x_{n+1}} \cdots \sum_{x_N} \frac{1}{Z} \prod_C \psi_C(\mathbf{x}_C)
 \end{aligned}$$

- This would be slow – $O(K^N)$ work if each variable could take K values

Inference on a Chain



- However, due to the factorization terms in this summation can be rearranged nicely
- This will lead to efficient algorithms

Simple Algebra

- This efficiency comes from a very simple distributivity

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

- Or more complicated version

$$\begin{aligned}\sum_i \sum_j a_i b_j &= a_1 b_1 + a_1 b_2 + \dots + a_n b_n \\ &= (a_1 + \dots + a_n)(b_1 + \dots + b_n)\end{aligned}$$

- Much faster to do right hand side ($2(n-1)$ additions, 1 multiplication) than left hand side (n^2 multiplications, $n^2 - 1$ additions)

A Simple Chain



- First consider a chain with 3 nodes, and computing $p(x_3)$:

$$\begin{aligned} p(x_3) &= \sum_{x_1} \sum_{x_2} \psi_{12}(x_1, x_2) \psi_{23}(x_2, x_3) \\ &= \sum_{x_2} \psi_{23}(x_2, x_3) \left[\sum_{x_1} \psi_{12}(x_1, x_2) \right] \end{aligned}$$

Performing the sums

$$p(x_3) = \sum_{x_2} \psi_{23}(x_2, x_3) \left[\sum_{x_1} \psi_{12}(x_1, x_2) \right]$$

- For example, if x_i are binary:

$$\psi_{12}(x_1, x_2) = x_1 \underbrace{\begin{bmatrix} a & b \\ c & d \end{bmatrix}}_{x_2} \quad \psi_{23}(x_2, x_3) = x_2 \underbrace{\begin{bmatrix} s & t \\ u & v \end{bmatrix}}_{x_3}$$

$$\sum_{x_1} \psi_{12}(x_1, x_2) = \underbrace{\begin{bmatrix} a + c & b + d \end{bmatrix}}_{x_2} \equiv \mu_{12}(x_2)$$

$$\psi_{23}(x_2, x_3) \times \mu_{12}(x_2) = x_2 \underbrace{\begin{bmatrix} s(a + c) & t(a + c) \\ u(b + d) & v(b + d) \end{bmatrix}}_{x_3}$$

$$p(x_3) = \begin{bmatrix} s(a + c) + u(b + d) & t(a + c) + v(b + d) \end{bmatrix}$$

Complexity of Inference

- There were two types of operations
 - Summation

$$\sum_{x_1} \psi_{12}(x_1, x_2)$$

$K \times K$ numbers in ψ_{12} , takes $O(K^2)$ time

- Multiplication

$$\psi_{23}(x_2, x_3) \times \mu_{12}(x_2)$$

Again $O(K^2)$ work

- For a chain of length N , we repeat these operations $N - 1$ times each
 - $O(NK^2)$ work, versus $O(N^K)$ for naive evaluation

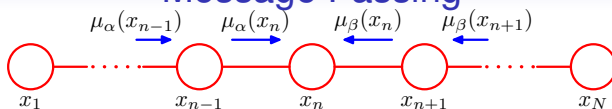
More complicated chain

- Now consider a 5 node chain, again asking for $p(x_3)$

$$\begin{aligned}
 p(x_3) &= \sum_{x_1} \sum_{x_2} \sum_{x_4} \sum_{x_5} \psi_{12}(x_1, x_2) \psi_{23}(x_2, x_3) \psi_{34}(x_3, x_4) \psi_{45}(x_4, x_5) \\
 &= \sum_{x_2} \sum_{x_1} \psi_{12}(x_1, x_2) \psi_{23}(x_2, x_3) \sum_{x_4} \sum_{x_5} \psi_{34}(x_3, x_4) \psi_{45}(x_4, x_5) \\
 &= \left[\sum_{x_2} \sum_{x_1} \psi_{12}(x_1, x_2) \psi_{23}(x_2, x_3) \right] \left[\sum_{x_4} \sum_{x_5} \psi_{34}(x_3, x_4) \psi_{45}(x_4, x_5) \right]
 \end{aligned}$$

- Each of these factors resembles the previous, and can be computed efficiently
 - Again $O(NK^2)$ work

Message Passing



- The factors can be thought of as **messages** being passed between nodes in the graph

$$\mu_{12}(x_2) \equiv \sum_{x_1} \psi_{12}(x_1, x_2)$$

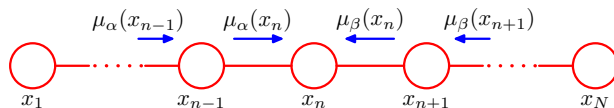
is a message passed from node x_1 to node x_2 containing all information in node x_1

- In general,

$$\mu_{k-1,k}(x_k) = \sum_{x_{k-1}} \psi_{k-1,k}(x_{k-1}, x_k) \mu_{k-2,k-1}(x_{k-1})$$

- Possible to do so because of conditional independence

Computing All Marginals



- Computing one marginal $p(x_n)$ takes $O(NK^2)$ time
- Naively running same algorithms for all nodes in a chain would take $O(N^2K^2)$ time
- But this isn't necessary, same messages can be reused at all nodes in the chain
- Pass all messages from one end of the chain to the other, pass all messages in the other direction too
- Can compute marginal at any node by multiplying the two messages delivered to the node
 - $2(N-1)K^2$ work, twice as much as for just one node

Including Evidence

- If a node $x_{k-1} = e$ is observed, simply clamp to observed value rather than summing:

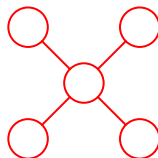
$$\mu_{k-1,k}(x_k) = \sum_{x_{k-1}} \psi_{k-1,k}(x_{k-1}, x_k) \mu_{k-2,k-1}(x_{k-1})$$

becomes

$$\mu_{k-1,k}(x_k) = \psi_{k-1,k}(x_{k-1} = e, x_k) \mu_{k-2,k-1}(x_{k-1} = e)$$

Trees

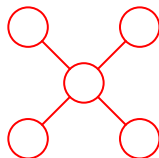
- The algorithm for a tree-structured graph is very similar to that for chains
- Formulation in PRML uses **factor graphs**, we'll just give the intuition here
- Consider calculating the marginal $p(x_n)$ for the center node of the graph at right
- Treat x_n as root of tree, pass messages from leaf nodes up to root



Trees

- Message passing similar to that in chains, but possibly multiple messages reaching a node
- With multiple messages, multiply them together
- As before, sum out variables

$$\mu_{k-1,k}(x_k) = \sum_{x_{k-1}} \psi_{k-1,k}(x_{k-1}, x_k) \mu_{k-2,k-1}(x_{k-1})$$



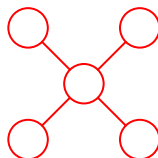
- Known as **sum-product algorithm**
- Complexity still $O(NK^2)$

Most Likely Configuration

- A similar algorithm exists for finding

$$\arg \max_{x_1, \dots, x_N} p(x_1, \dots, x_N)$$

- Replace summation operations with **maximize** operations
- Maximum of products at each node
- Known as **max-sum**, since often take log-probability to avoid underflow errors



General Graphs

- **Junction tree algorithm** is an exact inference method for arbitrary graphs
 - A particular tree structure defined over cliques of variables
 - Inference ends up being exponential in maximum clique size
 - Therefore slow in many cases
- Approximate inference techniques
 - **Loopy belief propagation**: run message passing scheme (sum-product) for a while
 - Sometimes works
 - Not guaranteed to converge
 - **Variational methods**: approximate desired distribution using analytically simple forms, find parameters to make these forms similar to actual desired distribution (Ch. 10, we won't cover)
 - **Sampling methods**: represent desired distribution with a set of samples, as more samples are used, obtain more accurate representation (Ch. 11, cancelled due to snow day)