Lecture 5
Graphical Models

University of Amsterdam
1 Introduction

2 Bayesian Networks
   - Independence
   - D-separation

3 Markov Random Fields
   - Independence properties
   - Factorisation

4 Factor Graphs
   - The basics
   - Conversions

5 Summing up
   - Graphical models as filters
   - Bayesian nets vs. Markov Random Fields vs. Factor Graphs

6 Inference
   - The sum-product algorithm
   - The max-sum algorithm
1 Introduction

2 Bayesian Networks
   - Independence
   - D-separation

3 Markov Random Fields
   - Independence properties
   - Factorisation

4 Factor Graphs
   - The basics
   - Conversions

5 Summing up
   - Graphical models as filters
   - Bayesian nets vs. Markov Random Fields vs. Factor Graphs

6 Inference
   - The sum-product algorithm
   - The max-sum algorithm
Probabilistic modelling

When given the joint probability distribution, we can answer any question about variables.

**Example**

If we know $p(A, B, C)$, we can answer questions such as $p(A|C)$, the probability that $A$ should have a certain value if $C$ is observed, using Bayes’ rule:

$$p(A|C) = \frac{p(A, C)}{p(C)}$$

where $p(A, C) = \int p(A, B, C) dB$ and $p(C) = \int\int p(A, B, C) dA dB$.
This requires **marginalisation**

- in general: exponential in number of variables
- computationally expensive or even intractable!
- complexity reduced if some variables are independent of others
- Graphical models provide a simple way to express independence
Probabilistic Graphical Models

Gained increasing popularity in Machine Learning because:

- They provide a simple way to visualise the structure of a probabilistic model and can be used to design and motivate new models.
- Insights into the property of the models can be obtained by inspection of the graph.
- Complex computations, required to perform inference and learning in sophisticated models, can be expressed in terms of graphical manipulations.
The basics

In a graphical model

- Random Variables are denoted as nodes, and they can be discrete or continuous
- Relations are denoted by edges (can be directed or undirected)
- Shaded nodes represent observed variables
- Plates represent repetition
The basics

In a graphical model

- Random Variables are denoted as nodes, and they can be discrete or continuous
- Relations are denoted by edges (can be directed or undirected)
- Shaded nodes represent observed variables
- Plates represent repetition
The basics

In a graphical model

- Random Variables are denoted as nodes, and they can be discrete or continuous
- Relations are denoted by edges (can be directed or undirected)
- Shaded nodes represent observed variables
- Plates represent repetition
The basics

In a graphical model

- Random Variables are denoted as nodes, and they can be discrete or continuous
- Relations are denoted by edges (can be directed or undirected)
- Shaded nodes represent observed variables
- Plates represent repetition

\[
\begin{align*}
T & \quad \rightarrow \\
& \quad A
\end{align*}
\]
The basics

In a graphical model
- Random Variables are denoted as nodes, and they can be discrete or continuous
- Relations are denoted by edges (can be directed or undirected)
- Shaded nodes represent observed variables
- Plates represent repetition
The basics

In a graphical model
- Random Variables are denoted as nodes, and they can be discrete or continuous
- Relations are denoted by edges (can be directed or undirected)
- Shaded nodes represent observed variables
- Plates represent repetition
The basics

In a graphical model

- Random Variables are denoted as nodes, and they can be discrete or continuous
- Relations are denoted by edges (can be directed or undirected)
- Shaded nodes represent observed variables
- Plates represent repetition

- The graphical model represents the factorisation of the joint distribution of the variables
- To use the model, we need to be able to do both learning and inference. In this lecture we focus on inference
1. Introduction

2. Bayesian Networks
   - Independence
   - D-separation

3. Markov Random Fields
   - Independence properties
   - Factorisation

4. Factor Graphs
   - The basics
   - Conversions

5. Summing up
   - Graphical models as filters
   - Bayesian nets vs. Markov Random Fields vs. Factor Graphs

6. Inference
   - The sum-product algorithm
   - The max-sum algorithm
Bayesian Networks

In this example we see nodes \( x = x_1 \ldots x_7 \).

- Their joint probability is 
  \( p(x) = p(x_1, x_2, \ldots, x_7) \).

- The graph implies an explicit factorisation of this joint distribution
  \( p(x) = \prod_{k=1}^{7} p(x_k|\text{pa}(x_k)) \).

\[
p(x) = p(x_1) p(x_2) p(x_3) p(x_4|x_1, x_2, x_3) p(x_5|x_1, x_3) p(x_6|x_4) p(x_7|x_4, x_5)
\]
Bayesian Networks

Example Bayesian Network

- In this example we see nodes $x = x_1 \ldots x_7$
- Their joint probability is $p(x) = p(x_1, x_2, \ldots, x_7)$
- The graph implies an explicit factorisation of this joint distribution
- $p(x) = \prod_{k=1}^{7} p(x_k | pa(x_k))$

$p(x) = p(x_1) p(x_2) p(x_3) p(x_4 | x_1, x_2, x_3) p(x_5 | x_1, x_3) p(x_6 | x_4) p(x_7 | x_4, x_5)$
Factorisation

The full joint distribution can always be factorised as

\[ p(x) = p(x_7|x_1, x_2, x_3, x_4, x_5, x_6) \cdot p(x_6|x_1, x_2, x_3, x_4, x_5) \cdot p(x_5|x_1, x_2, x_3, x_4) \cdot p(x_4|x_1, x_2, x_3) \cdot p(x_3|x_1, x_2) \cdot p(x_2|x_1) \cdot p(x_1) \]

for which we would need \(2^7 - 1\) parameters

\[ p(x) = p(x_1) \cdot p(x_2) \cdot p(x_3) \cdot p(x_4|x_1, x_2, x_3) \cdot p(x_5|x_1, x_3) \cdot p(x_6|x_4) \cdot p(x_7|x_4, x_5) \]

requires just 21 parameters.

- Remember: keep the simplest hypothesis that explains the data “well enough”
- Thus, the missing edges are what matters!
The full joint distribution can always be factorised as

\[ p(x) = p(x_7|x_1, x_2, x_3, x_4, x_5, x_6) \cdot p(x_6|x_1, x_2, x_3, x_4, x_5) \]
\[ \cdot p(x_5|x_1, x_2, x_3, x_4) \cdot p(x_4|x_1, x_2, x_3) \]
\[ \cdot p(x_3|x_1, x_2) \cdot p(x_2|x_1) \cdot p(x_1) \]

for which we would need \(2^7 - 1\) parameters

\[ p(x) = p(x_1) \cdot p(x_2) \cdot p(x_3) \cdot p(x_4|x_1, x_2, x_3) \cdot p(x_5|x_1, x_3) \cdot p(x_6|x_4) \cdot p(x_7|x_4, x_5) \]

requires just 21 parameters.

- Remember: keep the simplest hypothesis that explains the data “well enough”

- Thus, the missing edges are what matters!
Factorisation

The full joint distribution can always be factorised as

\[ p(x) = p(x_7|x_1, x_2, x_3, x_4, x_5, x_6) \cdot p(x_6|x_1, x_2, x_3, x_4, x_5) \]
\[ \cdot p(x_5|x_1, x_2, x_3, x_4) \cdot p(x_4|x_1, x_2, x_3) \]
\[ \cdot p(x_3|x_1, x_2) \cdot p(x_2|x_1) \cdot p(x_1) \]

for which we would need \(2^7 - 1\) parameters

\[ p(x) = \underbrace{p(x_1)}_{1} \cdot \underbrace{p(x_2)}_{1} \cdot \underbrace{p(x_3)}_{1} \cdot \underbrace{p(x_4|x_1, x_2, x_3)}_{8} \cdot \underbrace{p(x_5|x_1, x_3)}_{4} \cdot \underbrace{p(x_6|x_4)}_{2} \cdot \underbrace{p(x_7|x_4, x_5)}_{4} \]

requires just 21 parameters.

- Remember: keep the simplest hypothesis that explains the data “well enough”
- Thus, the missing edges are what matters!
Independence

Two sets of random variables $A$ and $B$ are *independent* (denoted as $A \perp \!\!\!\!\perp B$) if and only if

$$p(A, B) = p(A)p(B) \tag{1}$$

- The variables in set $A$ contain no information about those in set $B$. Learning the value(s) of variable(s) in set $A$, doesn’t change the probability distribution over the variables in set $B$.
- Imagine throwing two fair coins. Knowing that the first came heads, doesn’t change the distribution over the results of the second:

  $$ c_1 = H \quad c_1 = T $$
  $$ c_2 = H \quad 0.5 \quad 0.5 $$
  $$ c_2 = T \quad 0.5 \quad 0.5 $$

- From the product rule, eq. 1 implies that: $p(A|B) = p(A)$
- This provides no information about the *conditional* independence of variables.
Independence

Two sets of random variables $A$ and $B$ are independent (denoted as $A \perp \! \! \! \! \! \perp B$) if and only if

$$p(A, B) = p(A)p(B) \quad (1)$$

- The variables in set $A$ contain no information about those in set $B$. Learning the value(s) of variable(s) in set $A$, doesn’t change the probability distribution over the variables in set $B$.

- Imagine throwing two fair coins. Knowing that the first came heads, doesn’t change the distribution over the results of the second:

  $$c_1 = H \quad c_1 = T$$

  $$c_2 = H \quad 0.5 \quad 0.5$$

  $$c_2 = T \quad 0.5 \quad 0.5$$

- From the product rule, eq. 1 implies that: $p(A|B) = p(A)$

  - This provides no information about the conditional independence of variables
Two sets of random variables $A$ and $B$ are \textit{independent} (denoted as $A \perp \!\!\!\!\perp B$) if and only if

$$p(A, B) = p(A)p(B) \quad (1)$$

- The variables in set $A$ contain no information about those in set $B$. Learning the value(s) of variable(s) in set $A$, doesn’t change the probability distribution over the variables in set $B$.
- Imagine throwing two fair coins. Knowing that the first came heads, doesn’t change the distribution over the results of the second:

\[
\begin{array}{c|c|c}
   & c_1 = H & c_1 = T \\
\hline
 c_2 = H & 0.5 & 0.5 \\
 c_2 = T & 0.5 & 0.5 \\
\end{array}
\]

- From the product rule, eq. 1 implies that: $p(A|B) = p(A)$
- This provides no information about the \textbf{conditional} independence of variables
Conditional Independence

Two sets of random variables $A$ and $B$ are conditionally independent given a set $C$ if and only if

$$p(A, B|C) = p(A|C) p(B|C)$$

Here, the variables of set $A$ contain no information about those of set $B$ when we know the values of all the variables of set $C$.

Imagine throwing two fair coins, given the value of a function $f$ that indicates whether $c_1 = c_2$. Knowing that the first came heads, changes the distribution over the results of the second!

<table>
<thead>
<tr>
<th>$f$</th>
<th>$c_1$=H</th>
<th>$c_1$=T</th>
<th>$f$=1</th>
<th>$c_1$=H</th>
<th>$c_1$=T</th>
</tr>
</thead>
<tbody>
<tr>
<td>$c_2$=H</td>
<td>0</td>
<td>1</td>
<td>$c_2$=H</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$c_2$=T</td>
<td>1</td>
<td>0</td>
<td>$c_2$=T</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Similarly, equation 2 implies that: $p(A|C) = p(A|B, C)$

This is no information regarding any marginal independence between $A$ and $B$.

Intelligent Autonomous Systems
Conditional Independence

Two sets of random variables $A$ and $B$ are conditionally independent given a set $C$ if and only if

$$p(A, B|C) = p(A|C)p(B|C)$$

(2)

- Here, the variables of set $A$ contain no information about those of set $B$ when we know the values of all the variables of set $C$.

- Imagine throwing two fair coins, given the value of a function $f$ that indicates whether $c_1 = c_2$. Knowing that the first came heads, changes the distribution over the results of the second!

<table>
<thead>
<tr>
<th>$f=0$</th>
<th>$c_1=H$</th>
<th>$c_1=T$</th>
<th>$f=1$</th>
<th>$c_1=H$</th>
<th>$c_1=T$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$c_2=H$</td>
<td>0</td>
<td>1</td>
<td>$c_2=H$</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$c_2=T$</td>
<td>1</td>
<td>0</td>
<td>$c_2=T$</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

- Similarly, equation 2 implies that: $p(A|C) = p(A|B, C)$

- This is no information regarding any marginal independence between $A$ and $B$
Conditional Independence

Two sets of random variables $A$ and $B$ are conditionally independent given a set $C$ if and only if

$$p(A, B|C) = p(A|C) \cdot p(B|C)$$

(2)

- Here, the variables of set $A$ contain no information about those of set $B$ when we know the values of all the variables of set $C$.

- Imagine throwing two fair coins, given the value of a function $f$ that indicates whether $c_1 = c_2$. Knowing that the first came heads, changes the distribution over the results of the second!

<table>
<thead>
<tr>
<th>$f=0$</th>
<th>$c_1=H$</th>
<th>$c_1=T$</th>
<th>$f=1$</th>
<th>$c_1=H$</th>
<th>$c_1=T$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$c_2=H$</td>
<td>0</td>
<td>1</td>
<td>$c_2=H$</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$c_2=T$</td>
<td>1</td>
<td>0</td>
<td>$c_2=T$</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

- Similarly, equation 2 implies that: $p(A|C) = p(A|B, C)$

- This is no information regarding any marginal independence between $A$ and $B$
Enter college

Example

- Consider two characteristics of a person. Being smart, denoted by binary variable $S$, and being an athlete, denoted by binary variable $A$.
- Let’s assume that 40% of the population is smart, and 10% of the population is an athlete.
- Furthermore, let’s denote the fact that someone entered college with the binary variable $C$. If you are smart you have higher chances of entering college as well as if you are an athlete. Let’s say these probabilities are:

$$
\begin{array}{c|cc}
  & A = a & A = \neg a \\
\hline
 S = s & 0.91 & 0.90 \\
 S = \neg s & 0.90 & 0.04 \\
\end{array}
$$

- How would this graphical model look, and what would the factorisation imply?
Entering college

Example

\[ p(C, A, S) = p(C|A, S) \cdot p(A) \cdot p(S) \]

- What is the probability that an athlete is smart?
- What is the probability that a smart person is an athlete?
- Does this probability change if we meet this person in our college class?
Entering college

Example

\[ p(C, A, S) = p(C|A, S) p(A) p(S) \]

- What is the probability that an athlete is smart? \[0.4\]
- What is the probability that a smart person is an athlete?
- Does this probability change if we meet this person in our college class?
Bayesian Networks
Markov Random Fields
Factor Graphs
Summing up
Inference

Independence

Entering college

Example

\[ p(C, A, S) = p(C|A, S) p(A) p(S) \]

- What is the probability that an athlete is smart? \(0.4\)
- What is the probability that a smart person is an athlete? \(p(A|S) = 0.1\)
- Does this probability change if we meet this person in our college class?
Entering college

Example

\[ p(C, A, S) = p(C|A, S) p(A) p(S) \]

1. What is the probability that an athlete is smart? \(0.4\)
2. What is the probability that a smart person is an athlete? \(p(A|S) = 0.1\)
3. Does this probability change if we meet this person in our college class? \(p(A|S, C) \approx 0.1\)
Explaining away: an extreme example

Example

You want to pick up your bike which you locked close to central station. At central station, there are two reasons why bikes sometimes disappear:

1. It can be stolen
2. It can be vandalised, and the remnants cleaned up.

Let’s assume that $p(\text{gone} | \text{vandalised}) = 1$.

Questions:

- What is $p(\text{gone} | \text{stolen})$?
- If you notice your bike is gone, what happens to the probability that it was vandalised?
- What about $p(\text{stolen} | \text{gone})$?
- Now suppose you learn that it was stolen. What happens to $p(\text{vandalised} | \text{gone}, \text{stolen})$?
Explaining away: an extreme example

Example

You want to pick up your bike which you locked close to central station. At central station, there are two reasons why bikes sometimes disappear:

1. It can be stolen
2. It can be vandalised, and the remnants cleaned up.

Let’s assume that $p(\text{gone}|\text{vandalised}) = 1$.

Questions:

- What is $p(\text{gone}|\text{stolen})$?

$p(\text{gone}|\text{stolen}) = 1$

- If you notice your bike is gone, what happens to the probability that it was vandalised?

- What about $p(\text{stolen}|\text{gone})$?

- Now suppose you learn that it was stolen. What happens to $p(\text{vandalised}|\text{gone, stolen})$?
**Explaining away: an extreme example**

**Example**

You want to pick up your bike which you locked close to central station. At central station, there are two reasons why bikes sometimes disappear:

1. It can be stolen
2. It can be vandalised, and the remnants cleaned up.

Let’s assume that $p(\text{gone}|\text{vandalised}) = 1$.

Questions:

- What is $p(\text{gone}|\text{stolen})$?
  
  $p(\text{gone}|\text{stolen}) = 1$

- If you notice your bike is gone, what happens to the probability that it was vandalised?
  
  increases

- What about $p(\text{stolen}|\text{gone})$?

- Now suppose you learn that it was stolen. What happens to $p(\text{vandalised}|\text{gone, stolen})$?
Explaining away: an extreme example

Example

You want to pick up your bike which you locked close to central station. At central station, there are two reasons why bikes sometimes disappear:

1. It can be stolen
2. It can be vandalised, and the remnants cleaned up.

Let’s assume that $p(\text{gone} | \text{vandalised}) = 1$.

Questions:

- What is $p(\text{gone} | \text{stolen})$? $p(\text{gone} | \text{stolen}) = 1$
- If you notice your bike is gone, what happens to the probability that it was vandalised? increases
- What about $p(\text{stolen} | \text{gone})$? also increases
- Now suppose you learn that it was stolen. What happens to $p(\text{vandalised} | \text{gone}, \text{stolen})$?
Explaining away: an extreme example

Example

You want to pick up your bike which you locked close to central station. At central station, there are two reasons why bikes sometimes disappear:

1. It can be stolen
2. It can be vandalised, and the remnants cleaned up.

Let’s assume that \( p(\text{gone}|\text{vandalised}) = 1 \).

Questions:

- What is \( p(\text{gone}|\text{stolen}) \)? \( p(\text{gone}|\text{stolen}) = 1 \)
- If you notice your bike is gone, what happens to the probability that it was vandalised? increases
- What about \( p(\text{stolen}|\text{gone}) \)? also increases
- Now suppose you learn that it was stolen. What happens to \( p(\text{vandalised}|\text{gone, stolen}) \)? decreases
Detecting (conditional) independencies in the factorisation of a joint distribution is not easy.

- Independence of nodes in a graph can be found mechanically by operations on the graph.
- For the set of nodes $A$, $B$ and $C$,

$$A \perp \perp B \mid C$$ if all the paths from $A$ to $B$ are blocked.

- A path is blocked at a node when (d-separation)
  - edges meet head-to-tail ($\rightarrow\bigcirc\rightarrow$) or tail-to-tail ($\leftarrow\bigcirc\rightarrow$) at a node which is in the observed set $C$,
  - edges meet head-to-head ($\rightarrow\bigcirc\leftarrow$) at a node which is not in $C$, and none of whose descendents is in the observed set $C$. 
D-separation

A path is blocked at a node when (D-separation)

- edges meet head-to-tail (→〇→) or tail-to-tail (←〇→) in an observed node,

- edges meet head-to-head (→〇←) and the node nor any of its descendents is observed.
The Markov blanket of a node $x_i$:

- minimal set of nodes that “shield” the node $x_i$ from the rest of the graph
- Set of nodes, given which $x_i$ is independent from any other node in the graph
- For directed graphical models: set of parents, children and co-parents of the node
BayesNet Toolbox example

Example illustrating D-separation
Introduction

2 Bayesian Networks
   • Independence
   • D-separation

3 Markov Random Fields
   • Independence properties
   • Factorisation

4 Factor Graphs
   • The basics
   • Conversions

5 Summing up
   • Graphical models as filters
   • Bayesian nets vs. Markov Random Fields vs. Factor Graphs

6 Inference
   • The sum-product algorithm
   • The max-sum algorithm
The Basics

- Undirected graphical models are also known as Markov Random Fields or Markov networks.
- Each node corresponds to a variable or a group of variables.
- Edges denote relationships between variables.
Independence in MRFs

- We start by the independences a MRF represents, because they are easy to define.
- Once more, for the set of nodes $A, B$ and $C$, $A \perp \perp B \mid C$ if all the paths from $A$ to $B$ are blocked.
- A path from $A$ to $B$ is blocked when one of the path nodes belongs to set $C$. 
Independence in MRFs

An example where $A \perp B \mid C$ in an undirected graph
The Markov blanket of a (set of) nodes:

- Minimal set of nodes given which the nodes are independent of the rest of the graph
- No “explaining away”
- Markov blanket: set of neighbouring nodes
In this example we see nodes \( x = x_1, \ldots, x_4 \).

Independence between two nodes \( x_i \) and \( x_j \) corresponds to:

\[
p(x_i, x_j | x_{\setminus i,j}) = p(x_i | x_{\setminus i,j})p(x_j | x_{\setminus i,j})
\]

where \( x_{\setminus i,j} \) represents all the nodes in \( x \) except \( x_i \) and \( x_j \).

- **Clique** is a subset of a graph such that there exists a link between all pairs of nodes of the graph.
- **Maximal Clique** is a subset of a graph such that no other node can be added without it ceasing to be a clique.
The joint distribution of all the graph nodes can be written as a product of potential functions, each associated with a clique

\[ p(x) = \frac{1}{Z} \prod_C \psi_C(x_C) \]

where \( x_C \) are the nodes of the subset of clique \( C \), and \( Z \) the normalisation constant, usually called partition function, given by:

\[ Z = \sum_x \prod_C \psi_C(x_C) \]
Potential Functions

- They are non-negative
- They do not require a specific probabilistic interpretation
- That’s why we need an explicit normalisation term, which is sometimes *intractable* to compute!
- Comparison of different variable settings is easy
- Objective evaluation of a particular setting hard
Image Denoising

Example

- We represent the problem of image denoising with an undirected graphical model. Nodes $y_i$ represent observed pixel values, while nodes $x_i$ represent the unknowns and are the true pixel value in a noise-free image.
- Which are the maximal cliques of this model?
**Energy Function**

**Example**

- The nodes are binary and can take values $-1$ or $+1$
- We set $\eta$ as the potential of each clique $\{x_i, y_i\}$
- We set $\beta$ as the potential of each clique $\{x_i, x_j\}$
- We use $h$ to bias the model towards pixel values of a specific sign

**Energy function:**

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

**Potentials:**

$$p(x, y) = \frac{1}{Z} \exp(h \sum_i x_i - \beta \sum_{\{i, j\}} x_i x_j - \eta \sum_i x_i y_j)$$

$$= \frac{1}{Z} \psi_1(x)^h \psi_2(x)^{-\beta} \psi_3(x, y)^{-\eta}$$
Energy Function

Example

- The nodes are binary and can take values $-1$ or $+1$
- We set $\eta$ as the potential of each clique $\{x_i, y_i\}$
- We set $\beta$ as the potential of each clique $\{x_i, x_j\}$
- We use $h$ to bias the model towards pixel values of a specific sign
- Energy function:

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

- Potentials:

$$p(x, y) = \frac{1}{Z} \exp(h \sum_i x_i - \beta \sum_{\{i,j\}} x_i x_j - \eta \sum_i x_i y_j)$$

$$= \frac{1}{Z} \psi_1(x)^h \psi_2(x)^{-\beta} \psi_3(x, y)^{-\eta}$$
Energy Function

Example

- The nodes are binary and can take values $-1$ or $+1$
- We set $\eta$ as the potential of each clique $\{x_i, y_i\}$
- We set $\beta$ as the potential of each clique $\{x_i, x_j\}$
- We use $h$ to bias the model towards pixel values of a specific sign
- Energy function:

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

- Potentials:

$$p(x, y) = \frac{1}{Z} \exp(h \sum_i x_i - \beta \sum_{\{i,j\}} x_i x_j - \eta \sum_i x_i y_j)$$

$$= \frac{1}{Z} \psi_1(x)^h \psi_2(x)^{-\beta} \psi_3(x, y)^{-\eta}$$
Example: Iterated conditional modes

- We would like to infer the value of the variables $x_i$.
- We initially set $x_i = y_i$
- We observe each variable independently
- We change its value if this would increase the total configuration probability
  - We stop once we have iterated over all the variables without any value change
  - This will converge to a *local* optimum in the configuration space
Example: Iterated conditional modes

- We would like to infer the value of the variables $x_i$.
- We initially set $x_i = y_i$
- We observe each variable independently
- We change its value if this would increase the total configuration probability
- We stop once we have iterated over all the variables without any value change
- This will converge to a *local* optimum in the configuration space
Example: Iterated conditional modes

- We would like to infer the value of the variables $x_i$.
- We initially set $x_i = y_i$
- We observe each variable independently
- We change its value if this would increase the total configuration probability
- We stop once we have iterated over all the variables without any value change
- This will converge to a local optimum in the configuration space
1. Introduction

2. Bayesian Networks
   - Independence
   - D-separation

3. Markov Random Fields
   - Independence properties
   - Factorisation

4. Factor Graphs
   - The basics
   - Conversions

5. Summing up
   - Graphical models as filters
   - Bayesian nets vs. Markov Random Fields vs. Factor Graphs

6. Inference
   - The sum-product algorithm
   - The max-sum algorithm
### A factor graph

- In this example we see nodes $\mathbf{x} = x_1, \ldots, x_3$
- The joint distribution will be factored as:

$$p(x_1, x_2, x_3) = f_a(x_1, x_2) f_b(x_1, x_2) f_c(x_2, x_3) f_d(x_3)$$

- Which of these factors would be grouped together in an undirected graph?
- Does this provide more or less expressive power?
Undirected to Factor graph

Example

\[ x_1 \quad x_2 \quad x_3 \]

\[ x_1 \quad f \quad x_2 \]

\[ x_1 \quad f_a \quad x_2 \]

\[ x_1 \quad f_b \quad x_2 \]

Intelligent Autonomous Systems
Directed to Factor graph

Example

\[
\begin{align*}
\mathbf{x}_1 & \quad \mathbf{x}_2 & \quad \mathbf{x}_3 \\
\mathbf{x}_1 & \quad \mathbf{f} & \quad \mathbf{x}_2 \\
\mathbf{x}_1 & \quad \mathbf{f}_c & \quad \mathbf{x}_2 \\
\end{align*}
\]
<table>
<thead>
<tr>
<th>1. Introduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>2. Bayesian Networks</td>
</tr>
<tr>
<td>● Independence</td>
</tr>
<tr>
<td>● D-separation</td>
</tr>
<tr>
<td>3. Markov Random Fields</td>
</tr>
<tr>
<td>● Independence properties</td>
</tr>
<tr>
<td>● Factorisation</td>
</tr>
<tr>
<td>4. Factor Graphs</td>
</tr>
<tr>
<td>● The basics</td>
</tr>
<tr>
<td>● Conversions</td>
</tr>
<tr>
<td>5. <strong>Summing up</strong></td>
</tr>
<tr>
<td>● Graphical models as filters</td>
</tr>
<tr>
<td>● Bayesian nets vs. Markov Random Fields vs. Factor Graphs</td>
</tr>
<tr>
<td>6. Inference</td>
</tr>
<tr>
<td>● The sum-product algorithm</td>
</tr>
<tr>
<td>● The max-sum algorithm</td>
</tr>
</tbody>
</table>
Graphical models as filters

- Let \( p(x) \) be the set of all possible distributions over the variables at hand.
- Each graphical model is a filter for these distributions.
- Allowing only distributions that satisfy the appropriate factorisations go through.
Some factorisations can be expressed with a directed or undirected graph

Some can only be expressed with one of the two conventions

The factor graphs can express any kind of factorisation
1. Introduction

2. Bayesian Networks
   - Independence
   - D-separation

3. Markov Random Fields
   - Independence properties
   - Factorisation

4. Factor Graphs
   - The basics
   - Conversions

5. Summing up
   - Graphical models as filters
   - Bayesian nets vs. Markov Random Fields vs. Factor Graphs

6. Inference
   - The sum-product algorithm
   - The max-sum algorithm
The sum-product algorithm

- evaluates the local marginals over nodes or sets of nodes
- will be presented for discrete nodes. In the continuous case the sums become integrals
- is a more general case of an algorithm known as belief propagation
- is applicable on trees
Independence to simplify inference

If our variables are binary, the marginal $p(B)$ is:

$$p(B) = p(a, B, c) + p(a, B, \neg c) + p(\neg a, B, c) + p(\neg a, B, \neg c)$$

However, from our factorisation, we can simplify this as:

$$p(B) = p(a) p(B|a) [p(c|B) + p(\neg c|B)] + p(\neg a) p(B|\neg a) [p(c|B) + p(\neg c|B)]$$

$$= [p(a) p(B|a) + p(\neg a) p(B|\neg a)] [p(c|B) + p(\neg c|B)]$$

where we used that $(ab + ac) = a(b + c)$
The sum-product algorithm

Estimating $p(x)$

From the rules of probability

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

which under a factor graph becomes

$$p(x) = \sum_{x \setminus x} \prod_{s} f_s(x_s) = \sum_{x \setminus x} \prod_{s \in \text{ne}(x)} F_s(x, X_s)$$ (3)

where $\text{ne}(x)$ are the set of factor nodes that are neighbours of $x$

Essentially, we would like to explore the structure of the graph to

- obtain and efficient exact algorithm to obtain marginals
- in case we need several marginals, share the computations efficiently
We can substitute sums and products in eq 3:

\[
p(x) = \prod_{s \in \text{ne}(x)} \left[ \sum_{X_s} F_s(x, X_s) \right] = \prod_{s \in \text{ne}(x)} \mu_{f_s \rightarrow x}(x)
\]

where \( \mu_{f_s \rightarrow x}(x) \) can be viewed as a message from the factor node \( f_s \) to the variable \( x \).
Each message $\mu_{f_s \rightarrow x}(x)$ can be evaluated as:

$$\mu_{f_s \rightarrow x}(x) = \sum_{X_s} F_s(x, X_s)$$  \hspace{1cm} (4)

Each factor $F_s(x, X_s)$ is described by a new factor (sub-)graph where:

$$F_s(x, X_s) = f_s(x, x_1, x_2, \ldots, x_M) G_1(x_1, X_{s_1}) \cdots G_M(x_M, X_{s_M})$$  \hspace{1cm} (5)

where $x_1 \ldots x_M$ denote all the variables associated with $f_x$ but $x$. 
Substituting equation 5 in 4, we obtain:

\[
\mu_{f_s \to x}(x) = \sum_{x_1} \cdots \sum_{x_M} f_s(x, x_1, \ldots, x_M) \prod_{m \in \text{ne}(f_s) \setminus x} \left[ \sum_{X_{sm}} G_m(x_m, X_{sm}) \right]
\]

\[
= \sum_{x_1} \cdots \sum_{x_M} f_s(x, x_1, \ldots, x_M) \prod_{m \in \text{ne}(f_s) \setminus x} \mu_{x_m \to f_s}(x_m)
\]

where \(\mu_{x_m \to f_s}(x_m)\) can be viewed as a message from the variable \(x\) to the factor nodes \(f_s\)
In this case, $\mu_{x_m \rightarrow f_s}(x_m)$ is given by

$$
\mu_{x_m \rightarrow f_s}(x_m) = \sum_{x_{sm}} G_m(x_m, X_{sm}) \quad (6)
$$

with

$$
G_m(x_m, X_{sm}) = \prod_{l \in \text{ne}(x_m) \setminus f_s} F_l(x_m, X_{ml})
$$

If we substitute this in 6, we get

$$
\mu_{x_m \rightarrow f_s}(x_m) = \prod_{l \in \text{ne}(x_m) \setminus f_s} \left[ \sum_{x_{sm}} F_l(x_m, X_{ml}) \right] \\
= \prod_{l \in \text{ne}(x_m) \setminus f_s} \mu_{f_l \rightarrow x_m}(x_m)
$$
The algorithm

- We see node $x$ whose marginal we are after as the root of a tree.
- We start with messages from the leaves of the tree, 1 for nodes, $f(x)$ for factors.
- We compute the marginal when node $x$ receives all the incoming messages.
Example: Going to class

A  Attending class
B  Broken Bike
C  Consumption (of local products)
D  Despair (about succeeding for the class)
Example: Going to class

Probabilities:

\[ p(a|b, c) = 0 \quad p(b) = \frac{1}{12} \]
\[ p(a|b, \neg c) = \frac{1}{4} \quad p(c) = \frac{1}{3} \]
\[ p(a|\neg b, c) = \frac{1}{2} \quad p(d|a) = 0 \]
\[ p(a|\neg b, \neg c) = 1 \quad p(d|\neg a) = \frac{3}{4} \]
The sum-product algorithm

Example: Going to class

\[ f_a(B) = p(B) \]
\[ f_b(C) = p(C) \]
\[ f_c(A, B, C) = p(A|B, C) \]
\[ f_d(A, D) = p(D|A) \]
Example: Going to class

\[ \mu_1(D) = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \]

\[ \mu_2(B) = \begin{bmatrix} p(b) \\ p(\neg b) \end{bmatrix} = \begin{bmatrix} \frac{1}{12} \\ \frac{11}{12} \end{bmatrix} \]

\[ \mu_3(C) = \begin{bmatrix} p(c) \\ p(\neg c) \end{bmatrix} = \begin{bmatrix} \frac{1}{3} \\ \frac{2}{3} \end{bmatrix} \]
The sum-product algorithm

Example: Going to class

\[
\begin{align*}
\mu_4(A) &= \begin{bmatrix} 1p(d|a) + 1p(\neg d|a) \\ 1p(d|\neg a) + 1p(\neg d|\neg a) \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \\
\mu_5(B) &= \begin{bmatrix} p(b) \\ p(\neg b) \end{bmatrix} = \begin{bmatrix} \frac{1}{12} \\ \frac{11}{12} \end{bmatrix} \\
\mu_6(C) &= \begin{bmatrix} p(c) \\ p(\neg c) \end{bmatrix} = \begin{bmatrix} \frac{1}{3} \\ \frac{2}{3} \end{bmatrix}
\end{align*}
\]
Example: Going to class

\[ \mu_7(A) = \begin{bmatrix} p(b)p(c)p(a|b, c) + \cdots + p(\neg b)p(\neg c)p(a|\neg b, \neg c) \\ p(b)p(c)p(\neg a|b, c) + \cdots + p(\neg b)p(\neg c)p(\neg a|\neg b, \neg c) \end{bmatrix} \]

\[ = \begin{bmatrix} \frac{1}{12} \frac{1}{3} 1 + \frac{1}{12} \frac{2}{3} 4 + \frac{11}{12} \frac{1}{3} 2 + \frac{11}{12} \frac{2}{3} 1 \\ \frac{1}{12} \frac{1}{3} 1 + \frac{1}{12} \frac{2}{3} 4 + \frac{11}{12} \frac{1}{3} 2 + \frac{11}{12} \frac{2}{3} 1 \end{bmatrix} = \begin{bmatrix} \frac{2}{144} + \frac{22}{144} + \frac{88}{144} \\ \frac{4}{144} + \frac{6}{144} + \frac{22}{144} \end{bmatrix} = \begin{bmatrix} \frac{112}{144} \\ \frac{32}{144} \end{bmatrix} \]

\[ = \begin{bmatrix} \frac{7}{9} \\ \frac{2}{9} \end{bmatrix} = \begin{bmatrix} p(a) \\ p(\neg a) \end{bmatrix} \]
Example: Going to class

We can now compute the marginal probability at $A$:

$$\mu_4(A)\mu_7(A) = \begin{bmatrix} 1 & p(a) \\ 1 & p(\neg a) \end{bmatrix} = \begin{bmatrix} \frac{7}{9} \\ \frac{2}{9} \end{bmatrix}$$
Example: Going to class

\[ \mu_1(D) = \frac{1}{1} \]
\[ \mu_4(A) = \frac{1}{1} \]
\[ \mu_5(B) = \frac{1}{1} \]
\[ \mu_6(C) = \frac{1}{1} \]
\[ \mu_7(A) = \frac{1}{1} \]
\[ \mu_8(A) = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \]
\[ \mu_9(A) = \begin{bmatrix} p(a) \\ p(\neg a) \end{bmatrix} = \begin{bmatrix} 7/9 \\ 2/9 \end{bmatrix} \]
Example: Going to class

\[ \mu_{10}(D) = \begin{bmatrix} p(a) p(d|a) + p(\neg a) p(d|\neg a) \\ p(a) p(\neg d|a) + p(\neg a) p(\neg d|\neg a) \end{bmatrix} = \begin{bmatrix} p(d) \\ p(\neg d) \end{bmatrix} = \begin{bmatrix} \frac{7}{9} + \frac{2}{9} \\ \frac{1}{9} + \frac{4}{9} \end{bmatrix} = \begin{bmatrix} \frac{1}{6} \\ \frac{1}{6} \end{bmatrix} \]

\[ \mu_{11}(B) = \begin{bmatrix} p(a|b, c)p(c) + \cdots + p(\neg a|b, \neg c)p(\neg c) \\ p(a|\neg b, c)p(c) + \cdots + p(\neg a|\neg b, \neg c)p(\neg c) \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \mu_{12}(C) = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \]
Marginal over all nodes

- We can run the algorithm for each node independently
- In order to save time on computations we can have a full run over the whole factor graph
The max-sum algorithm

The most likely state of the system is not necessarily the state where all variables have their most likely state.

- We would like to acquire the most probable variable settings combination for our model.
- What would we acquire if we run the sum-product algorithm for each node of the graph, and set its value to

\[ x^* = \arg \max_x p(x) \]

- The max-sum algorithm estimates the node values that \textit{jointly} have the highest probability! That is:

\[ x^* = \arg \max_x p(x) \]
Maximising $p(x)$

We first write out the max operator in terms of its components:

$$\max_x p(x) = \max_{x_1} \max_{x_2} p(x) \cdot \max_{x_2} p(x) \cdots \max_{x_M} p(x)$$

which, given the factorisation provided by the factor graph and exchanging max operators and products becomes:

$$\max_x p(x) = \frac{1}{Z} \max_{x_1} \prod_{s \in \text{ne}(x_1)} F_s(x_1, X_s) \cdots \max_{x_M} \prod_{s \in \text{ne}(x_M)} F_s(x_M, X_s)$$

with all the terms having similar form to the sum-product algorithm.
max-sum messages

The messages to find the value of a node at the optimal joint configuration are:

\[ \mu_{f \rightarrow x} = \max_{x_1, x_2, \ldots, x_M} \left[ \ln f(x, x_1, \ldots, x_M) + \sum_{m \in \text{ne}(f_s) \setminus x} \mu_{x_m \rightarrow f}(x_m) \right] \]

where

\[ \mu_{x \rightarrow f}(x) = \sum_{l \in \text{ne}(x) \setminus f} \mu_{f_l \rightarrow x}(x) \]

Note the use of the logarithm to avoid computations with extremely small values! The products turn into sums, but the maximum remains.
The max-sum algorithm

With initialisations:

$$
\mu_{x \rightarrow f}(x) = 0 \text{ and } \mu_{f \rightarrow x}(x) = \ln f(x)
$$

at the root node we can compute the maximum probability as:

$$
p^{\text{max}} = \max_x \left[ \sum_{s \in \text{ne}(x)} \mu_{f_s \rightarrow x}(x) \right]
$$

and the node's value as:

$$
x^{\text{max}} = \arg \max_x \left[ \sum_{s \in \text{ne}(x)} \mu_{f_s \rightarrow x}(x) \right]
$$
Obtaining $x^{\text{max}}$ is not straightforward!

If we just propagate messages back, individual $x^*$ might correspond to different configuration values.

Instead we save these values as

$$
\phi(x_n) = \arg \max_{x_{n-1}} \left[ \ln f_{n-1,n}(x_{n-1}, x_n) + \mu_{x_{n-1} \rightarrow f_{n-1,n}} (x) \right]
$$

and then, when we have reached the root node

$$
x_{n-1}^{\text{max}} = \phi(x_{n}^{\text{max}})
$$
Incorporating evidence

How can we incorporate observations in the computation?

- The sum-product algorithm marginalises over all nodes in the graph.
- The sum is taken over all possible values for each variable.
- In order to include observations (Evidence), we want to compute the factors for the observed values only.
- Include an extra factor to the observed variables, that is one for the observed value and zero otherwise.
Graphical models provide a simple way to visualise the structure of a probabilistic model and complex computations can be expressed in terms of graphical manipulations.

We saw a general algorithm to perform inference in factor graphs

Reading: Bishop chapter 8 (8.1.(1,2,4), 8.4.(1,2))

Stay tuned, next week we will see how to learn the parameters of our Graphical Model!