Exact and approximate inference in graphical models
Régis Sabbadin, Nathalie Peyrard, Marie-Josée Cros, Simon de Givry, Alain Franc, Stephane Robin, Thomas Schiex, Matthieu Vignes

To cite this version:
Régis Sabbadin, Nathalie Peyrard, Marie-Josée Cros, Simon de Givry, Alain Franc, et al.. Exact and approximate inference in graphical models. 10th International Conference on Scalable Uncertainty Management (SUM), Oct 2017, Granada, Spain. hal-02116334

HAL Id: hal-02116334
https://hal.science/hal-02116334
Submitted on 5 Jun 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers. L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Exact and Approximate Inference in Graphical Models
Variable Elimination and beyond

Nathalie Peyrard\textsuperscript{a}, Marie-José Cros\textsuperscript{a}, Simon de Givry\textsuperscript{a}, Alain Franc\textsuperscript{b},
Stéphane Robin\textsuperscript{c},
Régis Sabbadin\textsuperscript{a}, Thomas Schiex\textsuperscript{a}, Matthieu Vignes\textsuperscript{a, d}

\textsuperscript{a} INRA UR 875 MIAT, Toulouse, France
\textsuperscript{b} INRA UMR 1202, BioGeCo, Bordeaux, France
\textsuperscript{c} INRA-AgroParisTech, UMR 518 MIA, Paris, France
\textsuperscript{d} IFS, Massey University, Palmerston North, New Zealand

SUM’17, Granada, Spain
October, 5, 2017
A graphical model over $\Lambda = \prod_{i=1}^{n} \Lambda_i$ is a decomposable function: $\psi: \prod_i \Lambda_i \rightarrow \mathbb{R}^+$ (the $\Lambda_i$ are finite) which writes, $\forall x \in \Lambda$:

$$
\psi(x) = \bigcirc_{B \in \mathcal{B}} \psi_B(x_B),
$$

where $\mathcal{B}$ is a set of subsets of $V = \{1, \ldots, n\}$ and $\bigcirc \in \{\prod, \sum, \text{min}, \text{max} \ldots\}$ is a combination operator.
Graphical models

- A graphical model over $\Lambda = \prod_{i=1}^{n} \Lambda_i$ is a decomposable function:
  $\psi : \prod \Lambda_i \rightarrow \mathbb{R}^+$ (the $\Lambda_i$ are finite) which writes, $\forall x \in \Lambda$:

  $$\psi(x) = \otimes_{B \in \mathcal{B}} \psi_B(x_B),$$

  where $\mathcal{B}$ is a set of subsets of $V = \{1, \ldots, n\}$ and $\otimes \in \{\prod, \sum, \text{min}, \text{max} \ldots\}$ is a combination operator.

- Used to model as well:
  - **Uncertainty**: (Hidden) Markov Chains, Bayesian networks, Markov Random Fields, Gaussian Graphical models, Possibilistic networks...
  - **Preferences**: (Weighted) Constraint Satisfaction Problems, Cost Function Networks...
  - **Both uncertainty and preferences**: Influence Diagrams...
Graphical models

- A graphical model over $\Lambda = \prod_{i=1}^{n} \Lambda_i$ is a decomposable function: $\psi : \prod_i \Lambda_i \to \mathbb{R}^+$ (the $\Lambda_i$ are finite) which writes, $\forall x \in \Lambda$:
  \[
  \psi(x) = \bigcirc_{B \in \mathcal{B}} \psi_B(x_B),
  \]
  where $\mathcal{B}$ is a set of subsets of $V = \{1, \ldots, n\}$ and $\bigcirc \in \{\prod, \sum, \min, \max \ldots\}$ is a combination operator.

- Used to model as well:
  - **Uncertainty**: (Hidden) Markov Chains, Bayesian networks, Markov Random Fields, Gaussian Graphical models, Possibilistic networks...
  - **Preferences**: (Weighted) Constraint Satisfaction Problems, Cost Function Networks...
  - **Both uncertainty and preferences**: Influence Diagrams...

- In numerous application domains:
  - **Reasoning**: causal inference, information extraction...
  - **Computers**: Computer vision, speech recognition, LDPC codes...
  - **Bioinformatics**: Gene regulatory networks, protein structure...
  - **Environmental modelling**: Spatial and spatiotemporal processes...
1. Graphical models

2. Inference tasks

3. Variable elimination, elimination ordering, treewidth

4. Message passing
Examples of probabilistic graphical models (1)

Definition (Markov chain)

\[ X = (X_1, \ldots, X_n) \] is a set of variables, with finite domains \( \{\Lambda_i\}_{i=1,\ldots,n} \).

\[
P(x_1, \ldots, x_n) = \underbrace{P(x_1)}_{\psi_1(x_1)} \times \underbrace{P(x_2|x_1)}_{\psi_{12}(x_1,x_2)} \times \ldots \times \underbrace{P(x_n|x_{n-1})}_{\psi_{(n-1)n}(x_{n-1},x_n)}
\]
Examples of probabilistic graphical models (1)

**Definition (Markov chain)**

- \( X = (X_1, \ldots, X_n) \) is a set of variables, with finite domains \( \{\Lambda_i\}_{i=1,\ldots,n} \).

\[
P(x_1, \ldots, x_n) = P(x_1) \times P(x_2|x_1) \times \ldots \times P(x_n|x_{n-1})
\]

\[
\psi_1(x_1) \psi_1(x_1,x_2) \ldots \psi_{n-1}(x_{n-1},x_n)
\]

**Definition (Hidden Markov chain)**

- \( X = (X_1, \ldots, X_n) \) is an *unobserved* Markov chain.
- \( Z = (Z_1, \ldots, Z_n) \) is a set of *observed* variables.
- \( P(z|x) = \prod_{i=1}^{n} P(z_i|x_i) \) (independent observations).

\[
P(x, z) = P(x_1) \times \left( \prod_{i=1}^{n-1} P(z_i|x_i) \times P(x_{i+1}|x_i) \right) \times P(z_n|x_n)
\]

\[
\psi_1(x_1) \psi_1(x_1,z_1) \psi_i(x_i,z_i) \psi_{i+1}(x_i,x_{i+1}) \psi_n(x_n,z_n)
\]
Examples of probabilistic graphical models (2)

Definition (Bayesian network)

- $X = (X_1, \ldots, X_n)$ is a set of variables, with finite domains $\{\Lambda_i\}_{i=1,\ldots,n}$.
- $\text{Par}(i) \subseteq \{1, \ldots, i - 1\}, \forall i = 2, \ldots, n$.

$$P(x_1, \ldots, x_n) = \underbrace{P(x_1)}_{\psi_1(x_1)} \times \prod_{i=2}^{n} \underbrace{P(x_i | x_{\text{Par}(i)})}_{\psi_{\text{Par}(i) \cup \{i\}}(x_i, x_{\text{Par}(i)})}$$
Examples of probabilistic graphical models (2)

Definition (Bayesian network)

- $X = (X_1, \ldots, X_n)$ is a set of variables, with finite domains $\{\Lambda_i\}_{i=1,\ldots,n}$.
- $\text{Par}(i) \subseteq \{1, \ldots, i - 1\}, \forall i = 2, \ldots, n$.

$$P(x_1, \ldots, x_n) = \frac{1}{\psi_0} \times \prod_{i=2}^{n} P(x_i | x_{\text{Par}(i)})$$

Definition (Markov Random Field)

- $G = (V, E)$ is an undirected graph with vertices $V = \{1, \ldots, n\}$, edges $E \in V \times V$ and $C$ is the set of cliques of $G$.
- $\{\psi_C : X_C \to \mathbb{R}^+\}_{C \in C}$ are strictly positive functions.

$$P(x_1, \ldots, x_n) = \frac{1}{\psi_0} \times \prod_{C \in C} \psi_C(x_C)$$
Examples of non-probabilistic graphical models

Definition (Possibilistic networks)

- \( X = (X_1, \ldots, X_n) \) is a set of variables, with finite domains \( \{\Lambda_i\}_{i=1,\ldots,n} \).
- \( \text{Par}(i) \subseteq \{1, \ldots, i-1\}, \forall i = 2, \ldots, n \).
- \( \pi(X_1, \ldots, X_n) \) takes values in a finite totally ordered scale \( L \).

\[
\pi(x_1, \ldots, x_n) = \min \left\{ \pi(x_1), \min_{i=2,\ldots,n} \pi(x_i | x_{\text{Par}(i)}) \right\}
\]

\[
\psi_1(x_1) \quad \psi_{\text{Par}(i) \cup \{i\}}(x_i, x_{\text{Par}(i)})
\]
Examples of non-probabilistic graphical models

Definition (Possibilistic networks)

- $X = (X_1, \ldots, X_n)$ is a set of variables, with finite domains $\{\Lambda_i\}_{i=1,\ldots,n}$.
- $\text{Par}(i) \subseteq \{1, \ldots, i - 1\}, \forall i = 2, \ldots, n$.
- $\pi(X_1, \ldots, X_n)$ takes values in a finite totally ordered scale $L$.

\[
\pi(x_1, \ldots, x_n) = \min \left\{ \pi(x_1), \min_{i=2,\ldots,n} \pi(x_i | x_{\text{Par}(i)}) \right\}
\]

\[
\psi_1(x_1) \bigcup \psi_{\text{Par}(i) \cup \{i\}}(x_i, x_{\text{Par}(i)})
\]

Definition (Cost Functions networks)

- $\{w_C : X_C \rightarrow \mathbb{R}^+\}_{C \in \mathcal{C}}$ are positive functions.

\[
w(x_1, \ldots, x_n) = \sum_{c \in \mathcal{C}} w_C(x_C)
\]
More complex graphical models

Some graphical models can have more than one combination operator:

<table>
<thead>
<tr>
<th>Markov decision process / Influence Diagram</th>
</tr>
</thead>
<tbody>
<tr>
<td>● Markov chain/Bayesian network <em>plus</em> decision variables <em>plus</em> cost functions network</td>
</tr>
<tr>
<td>● Expected utility function</td>
</tr>
</tbody>
</table>

→ We will limit ourselves to single operator graphical models!
Graphical model definition

Definition (Graphical model)

- Let $X = (X_1, \ldots, X_n)$ be a set of variables.
- $X_i$ takes values in $\Lambda_i \subseteq \mathbb{R}$.
- A realization of $X$ is denoted $x = (x_1, \ldots, x_n)$, with $x_i \in \Lambda_i$.
- A graphical model over $X$ is a function $\psi : \prod_i \Lambda_i \to \mathbb{R}$, which writes, for all $x \in X$:

$$\psi(x) = \bigcirc_{B \in \mathcal{B}} \psi_B(x_B),$$

where $\mathcal{B}$ is a set of subsets of $V = \{1, \ldots, n\}$, $\psi_B : \prod_{i \in B} \Lambda_i \to \mathbb{R}$ and $\bigcirc \in \{\prod, \sum, \min, \max \ldots\}$ is a combination operator.
Why are these models called “graphical”? 

They admit “graphical” representations:

- Useful for variables interactions vizualization
- Allow to directly extract computational features: Treewidth, perfect elimination ordering...
- Graphical properties sometimes exploited directly in algorithms
Why are these models called “graphical”?

Graphical representations of a graphical model:

- **Directed graph representation**
  ⇒ Represent conditional dependence between variables...

- **Undirected graph representation**
  ⇒ Represent conditional independence between variables...
  And useful to compute variable elimination orderings

- **Hypergraph representation**
  ⇒ Directly represents the graphical model functions scopes

- **Factor graph representation**
  ⇒ Equivalent to hypergraph representation.
  Useful in message passing algorithms
Illustration on a bayesian network

\[ P(X_1, \ldots, X_5) = P(X_1)P(X_2|X_1)P(X_3|X_1)P(X_4|X_1, X_3)P(X_5|X_2, X_4) \]
Illustration on a bayesian network

\[ P(X_1, \ldots, X_5) = P(X_1)P(X_2|X_1)P(X_3|X_1)P(X_4|X_1, X_3)P(X_5|X_2, X_4) \]
Illustration on a bayesian network

\[ P(X_1, \ldots, X_5) = P(X_1)P(X_2|X_1)P(X_3|X_1)P(X_4|X_1, X_3)P(X_5|X_2, X_4) \]
Illustration on a Bayesian network

\[ P(X_1, \ldots, X_5) = P(X_1)P(X_2|X_1)P(X_3|X_1)P(X_4|X_1, X_3)P(X_5|X_2, X_4) \]

\[ \psi_{12}(X_1, X_2) = P(X_1)P(X_2|X_1) \]

\[ \psi_{134}(X_1, X_3, X_4) = P(X_3|X_1)P(X_4|X_1, X_3) ; \psi_{245}(X_2, X_4, X_5) = P(X_5|X_2, X_4) \]
Illustration on a bayesian network

\[
P(X_1, \ldots, X_5) = P(X_1)P(X_2|X_1)P(X_3|X_1)P(X_4|X_1, X_3)P(X_5|X_2, X_4)
\]

\[
\psi_{12}(X_1, X_2) = P(X_1)P(X_2|X_1)
\]

\[
\psi_{134}(X_1, X_3, X_4) = P(X_3|X_1)P(X_4|X_1, X_3); \quad \psi_{245}(X_2, X_4, X_5) = P(X_5|X_2, X_4)
\]
1. Graphical models

2. Inference tasks

3. Variable elimination, elimination ordering, treewidth

4. Message passing
Inference tasks in graphical models

There are basically two usual inference tasks that can be solved using graphical models:

- Optimizing:

  **Most likely solutions** in stochastic graphical models

  \[
  x^\ast = \arg \max_x P(x_1, \ldots, x_n)
  \]

  **Most likely solutions** in possibilistic networks

  \[
  x^\ast = \arg \max_x \pi(x_1, \ldots, x_n)
  \]

  **min-cost solutions** in cost function networks

  \[
  x^\ast = \arg \min_x w(x_1, \ldots, x_n)
  \]
Inference tasks in graphical models

There are basically two usual inference tasks that can be solved using graphical models:

- Optimizing
- Counting:

### Marginal probabilities computation in stochastic graphical models

\[ P(x_B) = \sum_{x_{\bar{B}}} P(x_B, x_{\bar{B}}) \]

### Normalizing constant computation in Markov Random Fields

\[ Z = \sum_x \prod_{C \in C} \psi_C(x_C) \]

### Solution counting in constraint satisfaction problems

\[ \#SAT = |\{x, w(x) = 0\}| \]
Inference tasks in graphical models

There are basically two usual inference tasks that can be solved using graphical models:

- Optimizing
- Counting

These are sometimes interleaved, in more complex problems:

### Maximum Expected Utility computation in influence diagrams

\[
MEU = \max_{d_1, \ldots, d_m} \sum_{x_{S_1}} \ldots \max_{d_{D_k}} \sum_{x_{S_k}} P(x_1, \ldots, x_n \mid d_1, \ldots, d_m) U(x, d)
\]

### Generalized Quantified Boolean Formula satisfiability

\[
UnSAT = \min_{x_{A_1}} \max_{y_{B_1}} \ldots \min_{x_{A_k}} \max_{y_{B_k}} w(x_1, \ldots, x_n, y_1, \ldots, y_m)
\]

→ Some of the following inference approaches may be used as well for these more complex tasks!
Inference approaches

- **Exact inference**
  - Exhaustive exploration (optimizing, solution finding, counting)
  - Heuristic search (optimizing, solution finding)
  - Variable elimination (counting, optimizing, solution finding)

- **Approximate inference**
  - Sampling based approaches: counting, optimizing, both (e.g. reinforcement learning)...
  - Heuristic-based approaches
  - Loopy belief-propagation (approximate counting or optimizing)
  - Variational approximation (stochastic models)
1 Graphical models

2 Inference tasks

3 Variable elimination, elimination ordering, treewidth

4 Message passing
Variable elimination, case of Hidden Markov Chains

Problem
Computing the most likely values of variables $H$ given a realization $o$ of the variables $O$. The problem is to compute $\text{arg max}_h p(H = h | O = o)$, or equivalently the argument of:

$$\max_{h_1, \ldots, h_T} \psi_{H_1}(h_1) \left( \prod_{i=1}^{T-1} \psi_{H_i, H_{i+1}}(h_i, h_{i+1}) \psi_{O_i, H_i}(o_i, h_i) \right) \psi_{O_T, H_T}(o_T, h_T)$$

Remarks:

- The number of possible realizations of $H$ is exponential in $T$
- Nevertheless, this optimization problem can be solved in a number of operations linear in $T$ using the well-known Viterbi algorithm
Variable elimination in HMC: Viterbi algorithm

$$\max_{h_1, \ldots, h_T} \psi_{H_1}(h_1) \left( \prod_{i=1}^{T-1} \psi_{H_i, H_{i+1}}(h_i, h_{i+1}) \psi_{O_i, H_i}(o_i, h_i) \right) \psi_{O_T, H_T}(o_T, h_T)$$
Variable elimination in HMC: Viterbi algorithm

\[
= \max_{h_1, \ldots, h_{T-1}} \psi_{H_1}(h_1) \left( \prod_{i=1}^{T-2} \psi_{H_i, H_{i+1}}(h_i, h_{i+1}) \psi_{O_i, H_i}(o_i, h_i) \right) \\
\times \psi_{O_{T-1}, H_{T-1}}(o_{T-1}, h_{T-1}) \times \max_{h_T} \psi_{H_{T-1}, H_T}(h_{T-1}, h_T) \psi_{O_T, H_T}(o_T, h_T)
\]

New potential function \( \psi'_{O_T, H_{T-1}}(o_T, h_{T-1}) \)

Viterbi algorithm

Diagram of the Viterbi algorithm with nodes representing hidden states and observations.
Variable elimination in HMC: Viterbi algorithm

\[
\begin{align*}
\text{max} & \prod_{i=1}^{T-2} \psi_{H_i, H_{i+1}}(h_i, h_{i+1}) \psi_{O_i, H_i}(o_i, h_i) \\
\times & \psi_{O_{T-1}, H_{T-1}}(o_{T-1}, h_{T-1}) \times \text{max} \psi_{H_{T-1}, H_T}(h_{T-1}, h_T) \psi_{O_T, H_T}(o_T, h_T)
\end{align*}
\]

New potential function: \( \psi'_{O_T, H_{T-1}}(o_T, h_{T-1}) \)
Variable elimination, bayesian network

\[ P(X_1, \ldots, X_6) = P(X_1)P(X_3)P(X_6|X_1, X_3)P(X_2|X_6)P(X_4|X_6)P(X_5|X_2, X_4) \]
Variable elimination, bayesian network

\[ P(X_1, \ldots, X_6) = P(X_1)P(X_3)P(X_6 | X_1, X_3)P(X_2 | X_6)P(X_4 | X_6)P(X_5 | X_2, X_4) \]
\[ P(X_1, \ldots, X_6) = \psi_{136}(X_1, X_3, X_6)\psi_{26}(X_2, X_6)\psi_{46}(X_4, X_6)\psi_{245}(X_2, X_4, X_5) \]
\[ P(X_1, \ldots, X_6) = \psi_{136}\psi_{26}\psi_{46}\psi_{245} \]
Variable elimination, bayesian network

\[ P(X_1, \ldots, X_6) = P(X_1)P(X_3)P(X_6 | X_1, X_3)P(X_2 | X_6)P(X_4 | X_6)P(X_5 | X_2, X_4) \]

\[ P(X_1, \ldots, X_6) = \psi_{136}(X_1, X_3, X_6)\psi_{26}(X_2, X_6)\psi_{46}(X_4, X_6)\psi_{245}(X_2, X_4, X_5) \]

\[ P(X_1, \ldots, X_6) = \psi_{136}\psi_{26}\psi_{46}\psi_{245} \]

Directed graph

\[
\begin{align*}
1 & \rightarrow 2 \\
6 & \rightarrow 5 \\
3 & \rightarrow 4
\end{align*}
\]

Undirected graph / Hypergraph

Compute \( \alpha = \max_{X_1, \ldots, X_6} P(X_1, \ldots, X_6) = \max_{X_1, \ldots, X_6} \psi_{136}\psi_{26}\psi_{46}\psi_{245} \)
Elimination ordering

- Variable elimination, for elimination order \( \pi = \{X_6, X_5, X_4, X_3, X_2, X_1\} \):

\[
\alpha = \max_{X_1, \ldots, X_6} \psi_{136}(X_1, X_3, X_6) \psi_{26}(X_2, X_6) \psi_{46}(X_4, X_6) \psi_{245}(X_2, X_4, X_5)
\]

\[
\alpha = \max_{X_1, \ldots, X_5} \psi_{245} \max_{X_6} \psi_{136} \psi_{26} \psi_{46} = \max_{X_1, \ldots, X_5} \psi_{245} \psi'_{1234}
\]

\[
\alpha = \max_{X_1, \ldots, X_4} \psi'_{1234} \max_{X_5} \psi_{245} = \max_{X_1, \ldots, X_4} \psi'_{1234} \psi'_{24} = \max_{X_1, \ldots, X_4} \psi''_{1234}
\]
Elimination ordering

• Variable elimination, for elimination order
  \( \pi = \{X_6, X_5, X_4, X_3, X_2, X_1\} \):

  \[
  \alpha = \max_{X_1, \ldots, X_6} \psi_{136}(X_1, X_3, X_6) \psi_{26}(X_2, X_6) \psi_{46}(X_4, X_6) \psi_{245}(X_2, X_4, X_5)
  \]

  \[
  \alpha = \max_{X_1, \ldots, X_5} \psi_{245} \max_{X_6} \psi_{136} \psi_{26} \psi_{46} = \max_{X_1, \ldots, X_5} \psi_{245} \psi'_{1234}
  \]

  \[
  \alpha = \max_{X_1, \ldots, X_4} \psi'_{1234} \max_{X_5} \psi_{245} = \max_{X_1, \ldots, X_4} \psi'_{1234} \psi'_{24} = \max_{X_1, \ldots, X_4} \psi''_{1234}
  \]

⇒ \( \pi \) generates functions with 4 variables, instead of 3 at most for \( \pi^* \)!
Elimination ordering

- Variable elimination, for elimination order $\pi = \{X_6, X_5, X_4, X_3, X_2, X_1\}$:

$$\alpha = \max_{X_1, \ldots, X_6} \psi_{136}(X_1, X_3, X_6) \psi_{26}(X_2, X_6) \psi_{46}(X_4, X_6) \psi_{245}(X_2, X_4, X_5)$$

$$\alpha = \max_{X_1, \ldots, X_5} \psi_{245} \max_{X_6} \psi_{136} \psi_{26} \psi_{46} = \max_{X_1, \ldots, X_5} \psi_{245} \psi'_{1234}$$

$$\alpha = \max_{X_1, \ldots, X_4} \psi'_{1234} \max_{X_5} \psi_{245} = \max_{X_1, \ldots, X_4} \psi'_{1234} \psi'_{24} = \max_{X_1, \ldots, X_4} \psi''_{1234}$$

- Is there an elimination order generating “smaller scopes” functions?
Elimination ordering

- Variable elimination, for elimination order \( \pi = \{X_6, X_5, X_4, X_3, X_2, X_1\} \):

\[
\alpha = \max_{X_1, \ldots, X_6} \psi_{136}(X_1, X_3, X_6) \psi_{26}(X_2, X_6) \psi_{46}(X_4, X_6) \psi_{245}(X_2, X_4, X_5)
\]

\[
\alpha = \max_{X_1, \ldots, X_5} \psi_{245} \max_{X_6} \psi_{136} \psi_{26} \psi_{46} = \max_{X_1, \ldots, X_5} \psi_{245} \psi_{1234}'
\]

\[
\alpha = \max_{X_1, \ldots, X_4} \psi_{1234}' \max_{X_5} \psi_{245} = \max_{X_1, \ldots, X_4} \psi_{1234}' \psi_{24} = \max_{X_1, \ldots, X_4} \psi_{1234}'
\]

- Is there an elimination order generating “smaller scopes” functions?

- Yes! \( \pi^* = \{X_5, X_4, X_2, X_6, X_3, X_1\} \):

\[
\alpha = \max_{X_1, X_3, X_6, X_2, X_4, X_5} \psi_{136} \psi_{26} \psi_{46} \psi_{245}
\]

\[
\alpha = \max_{X_1, \ldots, X_4} \psi_{136} \psi_{26} \psi_{46} \max_{X_5} \psi_{245} = \max_{X_1, \ldots, X_4} \psi_{136} \psi_{26} \psi_{46} \psi_{24}'
\]

\[
\alpha = \max_{X_1, \ldots, X_2} \psi_{136} \max_{X_4} \psi_{246} = \max_{X_1, X_3, X_6} \psi_{136} \max_{X_2} \psi_{26} = \max_{X_1, X_3, X_6} \psi_{136}'
\]
Elimination ordering

Yes! \( \pi^* = \{X_5, X_4, X_2, X_6, X_3, X_1\} \):

\[
\alpha = \max_{X_1,X_3,X_6,X_2,X_4,X_5} \psi_{136} \psi_{26} \psi_{46} \psi_{245}
\]

\[
\alpha = \max_{X_1,\ldots,X_4} \psi_{136} \psi_{26} \psi_{46} \max_{X_5} \psi_{245} = \max_{X_1,\ldots,X_4} \psi_{136} \psi_{26} \psi_{46} \psi_{24}'
\]

\[
\alpha = \max_{X_1,\ldots,X_2} \psi_{136} \max_{X_4} \psi_{246}' = \max_{X_1,X_3,X_6} \psi_{136} \max_{X_2} \psi_{26}' = \max_{X_1,X_3,X_6} \psi_{136}'
\]
Yes! $\pi^* = \{X_5, X_4, X_2, X_6, X_3, X_1\}$:

$$\alpha = \max_{X_1, X_3, X_6, X_2, X_4, X_5} \psi_{136} \psi_{26} \psi_{46} \psi_{245}$$

$$\alpha = \max_{X_1, \ldots, X_4} \psi_{136} \psi_{26} \psi_{46} \max_{X_5} \psi_{245} = \max_{X_1, \ldots, X_4} \psi_{136} \psi_{26} \psi_{46} \psi_{24}$$

$$\alpha = \max_{X_1, \ldots, X_2} \psi_{136} \max_{X_4} \psi_{246} = \max_{X_1, X_3, X_6} \psi_{136} \max_{X_2} \psi_{26} = \max_{X_1, X_3, X_6} \psi_{136}$$
Yes! $\pi^* = \{X_5, X_4, X_2, X_6, X_3, X_1\}$:

$$\alpha = \max_{X_1, X_3, X_6, X_2, X_4, X_5} \psi_{136} \psi_{26} \psi_{46} \psi_{245}$$

$$\alpha = \max_{X_1, \ldots, X_4} \psi_{136} \psi_{26} \psi_{46} \max_{X_5} \psi_{245} = \max_{X_1, \ldots, X_4} \psi_{136} \psi_{26} \psi_{46} \psi'_{24}$$

$$\alpha = \max_{X_1, \ldots, X_2} \psi_{136} \max_{X_4} \psi'_{246} = \max_{X_1, X_3, X_6} \psi_{136} \max_{X_2} \psi'_{26} = \max_{X_1, X_3, X_6} \psi'_{136}$$
Elimination ordering

Yes! $\pi^* = \{X_5, X_4, X_2, X_6, X_3, X_1\}$:

\[
\alpha = \max_{X_1, X_3, X_6, X_2, X_4, X_5} \psi_{136} \psi_{26} \psi_{46} \psi_{245}
\]
\[
\alpha = \max_{X_1, \ldots, X_4} \psi_{136} \psi_{26} \psi_{46} \max_{X_5} \psi_{245} = \max_{X_1, \ldots, X_4} \psi_{136} \psi_{26} \psi_{46} \psi'_{24}
\]
\[
\alpha = \max_{X_1, \ldots, X_2} \psi_{136} \max_{X_4} \psi'_{246} = \max_{X_1, X_3, X_6} \psi_{136} \max_{X_2} \psi'_{26} = \max_{X_1, X_3, X_6} \psi'_{136}
\]

$\Rightarrow \pi$ generates functions with 4 variables, instead of 3 at most for $\pi^*$!
Treewidth

Let $\mathcal{H} = \{H_1, \ldots, H_m\}$ be the hypergraph of the graphical model.

**Definition (Variable/vertex elimination)**

Let $i \in \{1, \ldots, n\}$ be a vertex/variable number. We define:

$$\mathcal{H}^{\downarrow i} = \{H_j \in \mathcal{H}, i \not\in H_j\} \cup \left\{ \bigcup_{i \in H_j \in \mathcal{H}} H_j \setminus \{i\} \right\}. $$
Treewidth

Let $\mathcal{H} = \{H_1, \ldots, H_m\}$ be the hypergraph of the graphical model.

Definition (Variable/vertex elimination)

Let $i \in \{1, \ldots, n\}$ be a vertex/variable number. We define:

$$\mathcal{H}^\downarrow_i = \{H_j \in \mathcal{H}, i \notin H_j\} \cup \left\{ \bigcup_{i \in H_j \in \mathcal{H}} H_j \setminus \{i\} \right\}.$$ 

Definition (Induced width)

- Let $\pi$ be an elimination order ($\pi(k)$ is the $k^{th}$ eliminated vertex).
- Let $\mathcal{H}^k$ be the hypergraph generated after $k$ vertices eliminations ($\mathcal{H}^0 = \mathcal{H}$ and $\mathcal{H}^n = \emptyset$).

$\Rightarrow$ The induced width is the size of the largest generated hyperedge (minus 1), during the process:

$$\text{IWX}_\pi(\mathcal{H}) = \max_{k=0,\ldots,n-1} \max_{H \in \mathcal{H}^k} |H| - 1.$$
Treewidth

Let \( \mathcal{H} = \{ H_1, \ldots, H_m \} \) be the hypergraph of the graphical model.

**Definition (Treewidth)**

\( TW(\mathcal{H}) \), the **treewidth** of \( \mathcal{H} \) is the minimum induced width over all elimination orders:

\[
TW(\mathcal{H}) = \min_{\pi} IW_{\pi}(\mathcal{H}).
\]
Treewidth

Let $\mathcal{H} = \{H_1, \ldots, H_m\}$ be the hypergraph of the graphical model.

**Definition (Treewidth)**

$\mathcal{T}_W(\mathcal{H})$, the treewidth of $\mathcal{H}$ is the minimum induced width over all elimination orders:

$$\mathcal{T}_W(\mathcal{H}) = \min_{\pi} \mathcal{I}_W(\pi(\mathcal{H})).$$
Treewidth

Let $\mathcal{H} = \{H_1, \ldots, H_m\}$ be the hypergraph of the graphical model.

**Definition (Treewidth)**

$\mathcal{TW}(\mathcal{H})$, the treewidth of $\mathcal{H}$ is the minimum induced width over all elimination orders:

$$\mathcal{TW}(\mathcal{H}) = \min_{\pi} \mathcal{IW}_{\pi}(\mathcal{H}).$$

**Undirected graph / Hypergraph**

- $\pi = \{6, 5, 4, 3, 2, 1\}$
- $\mathcal{IW}_{\pi}(\mathcal{H}) = 3$
Treewidth

Let $\mathcal{H} = \{H_1, \ldots, H_m\}$ be the hypergraph of the graphical model.

**Definition (Treewidth)**

$\mathcal{T}W(\mathcal{H})$, the treewidth of $\mathcal{H}$ is the minimum induced width over all elimination orders:

$$\mathcal{T}W(\mathcal{H}) = \min_{\pi} \mathcal{I}W_{\pi}(\mathcal{H}).$$

**Undirected graph / Hypergraph**

- $\pi = \{6, 5, 4, 3, 2, 1\}$
- $\mathcal{I}W_{\pi}(\mathcal{H}) = 3$
- $\pi^* = \{5, 2, 4, 6, 3, 1\}$
- $\mathcal{T}W(\mathcal{H}) = \mathcal{I}W_{\pi^*}(\mathcal{H}) = 2$
Treewidth computation

Variable elimination takes exponential space/time in induced width

⇒ Important to find a “good” elimination order (small induced width)
Treewidth computation

Variable elimination takes exponential space/time in induced width
⇒ Important to find a “good” elimination order (small induced width)

Determining the treewidth is a hard problem
- Computing the treewidth is NP-hard
- Approximating the treewidth with constant factor?
⇒ Not known whether in P or NP-hard
Treewidth computation

Variable elimination takes exponential space/time in induced width
⇒ Important to find a “good” elimination order (small induced width)

Determining the treewidth is a hard problem
- Computing the treewidth is $\text{NP}$-hard
- Approximating the treewidth with constant factor?
⇒ Not known whether in $\text{P}$ or $\text{NP}$-hard

Treewidth computation is a Fixed Parameter Tractable problem
- An $O\left(\mathcal{TW}(\mathcal{H})^{\mathcal{TW}(\mathcal{H})^3} n\right)$ solution algorithm (impractical)
Treewidth computation

Variable elimination takes exponential space/time in induced width
⇒ Important to find a “good” elimination order (small induced width)

Determining the treewidth is a hard problem
- Computing the treewidth is NP-hard
- Approximating the treewidth with constant factor?
⇒ Not known whether in P or NP-hard

Treewidth computation is a Fixed Parameter Tractable problem
- An $O\left(\mathcal{T}\mathcal{W}(\mathcal{H})^{\mathcal{T}\mathcal{W}(\mathcal{H})^3} n\right)$ solution algorithm (impractical)

⇒ In practice, elimination order heuristics (e.g. min-fill).
1. Graphical models

2. Inference tasks

3. Variable elimination, elimination ordering, treewidth

4. Message passing
Message passing algorithms are powerful procedures for exact and approximate inference in graphical models:

- They are derived from variable elimination algorithms.
- They can be more powerful for exact inference:
  - Using twice as much work, they provide all marginals of a SGM, over all variables.
- They give either exact or approximate solutions:
  - **Exact** Message passing on a tree-graphical model $\sim$ variable elimination
  - **Exact** Message passing on a tree-factor graph graphical model
  - **Approx** Message passing in the general case: (generalized) loopy Belief Propagation (LBP)
Message passing on a tree

Example: Pairwise graphical model

- Messages computation mimics variable elimination
- For marginalization, message updates take the form:

\[
\forall x_j, \mu_{ij}(x_j) \leftarrow \frac{1}{K} \sum_{x_i} \psi_{ij}(x_i, x_j)\psi_i(x_i) \prod_{k \neq j, (i,j) \in E} \mu_{ki}(x_i)
\]
Once messages have been asynchronously computed along all edges in the two directions, marginal probabilities can be obtained:

\[ p_i(x_i) \leftarrow \frac{1}{K_i} \psi_i(x_i) \prod_{j, (j,i) \in E} \mu_{ji}(x_i), \forall x_i, \]

\[ p_{ij}(x_i, x_j) \leftarrow \frac{1}{K_{ij}} \psi_{ij}(x_{ij}) \prod_{k \neq j, (k,i) \in E} \mu_{ki}(x_i) \prod_{l \neq i, (l,j) \in E} \mu_{lj}(x_j). \]
Message passing on a tree factor graph

Example: Hypergraph and equivalent tree factor graph

- The factor graph is not a tree...
- But becomes one if we merge variables $X_2$ and $X_4$, to get a new variable $X'_24$. 
Now that we have a tree, two different kinds of messages are computed:

- Factor-to-variable messages: $\mu_{fi}(x_i)$.
- Variable-to-factor messages: $\nu_{if}(x_i)$. 

Example: Hypergraph and equivalent tree factor graph
Message passing on a tree factor graph

Example: Hypergraph and equivalent tree factor graph

Update rules:
\[
\mu_{fi}(x_i) \leftarrow \sum_{x_{f\setminus i}} \left( \psi_f(x_f) \prod_{j \in f \setminus i} \nu_{jf}(x_j) \right), \forall x_i,
\]
\[
\nu_{if}(x_i) \leftarrow \prod_{f' \neq f, i \in f'} \mu_{f'i}(x_i), \forall x_i.
\]
After two complete passes, marginal probabilities can be computed:

\[ p_i(x_i) \leftarrow \frac{1}{K} \psi_i(x_i) \prod_{f, i \in f} \mu_{fi}(x_i), \forall x_i. \]
General case: Loopy Belief Propagation

- When the graphical model is not a tree?
- When the factor graph is not a tree?

⇒ Message passing can still be applied, to approximate marginals!
General case: Loopy Belief Propagation

- When the graphical model is not a tree?
- When the factor graph is not a tree?

⇒ **Message passing** can still be applied, to **approximate marginals**!

### Loopy Belief Propagation

- Messages sent asynchronously along edges of the graph/factor graph.
- Until an arbitrary convergence condition is met
General case: Loopy Belief Propagation

- When the graphical model is not a tree?
- When the factor graph is not a tree?

⇒ Message passing can still be applied, to approximate marginals!

Loopy Belief Propagation

- Messages sent asynchronously along edges of the graph/factor graph.
- Until an arbitrary convergence condition is met

😢 Convergence to a steady-state is not guaranteed
😢 No guarantee on the “quality” of the approximated marginals
General case: Loopy Belief Propagation

- When the graphical model is not a tree?
- When the factor graph is not a tree?

⇒ **Message passing can still be applied, to approximate marginals!**

**Loopy Belief Propagation**

- Messages sent asynchronously along edges of the graph/factor graph.
- Until an arbitrary convergence condition is met

🤔 **Convergence to a steady-state is not guaranteed**
😢 **No guarantee on the “quality” of the approximated marginals**
😊 **LBP steady states have a variational approximation interpretation**
😀 **“Good” approximation in practice**
Summary and concluding remarks

**Graphical models**

- A convenient approach to model uncertainty and preferences
- A very general model, with variants (combination operator)
- Useful to solve several problems (optimizing, marginalizing...)

A generic algorithmic approach for exact inference (variable elimination) or approximate inference (LBP)
Summary and concluding remarks

Graphical models

- A convenient approach to model uncertainty and preferences
- A very general model, with variants (combination operator)
- Useful to solve several problems (optimizing, marginalizing…)

A generic algorithmic approach for exact inference (variable elimination) or approximate inference (LBP)

Other ways to perform inference in graphical models

- Search (maximizing)
- Monte-Carlo simulation (lots of theoretical and practical results)
- Mathematical programming…

Another important question: Infering structure of GM from data!
References

To read more about the topic of this talk:


And some very useful references:

