Principles of Managing Uncertain Data

Lecture 12: More on Probabilistic Databases
Table of Contents

1. Introduction
2. Markov Logic Networks
3. Probabilistic XML
Many thanks to Dan Suciu for advising on these slides!
Table of Contents

1 Introduction

2 Markov Logic Networks

3 Probabilistic XML
So far, we have learned about representations of probabilistic databases
So far, we have learned about representations of probabilistic databases

Focusing on tuple-independent databases (TID), we studied query evaluation in depth
Introduction

- So far, we have learned about representations of probabilistic databases
- Focusing on tuple-independent databases (TID), we studied query evaluation in depth
- This lecture: two additional notions of probabilistic databases:
  - Markov Logic Networks
  - Probabilistic XML
Table of Contents

1 Introduction

2 Markov Logic Networks

3 Probabilistic XML
Running Example

Follows

<table>
<thead>
<tr>
<th>p1</th>
<th>p2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anna</td>
<td>Bob</td>
</tr>
<tr>
<td>Bob</td>
<td>Chloe</td>
</tr>
<tr>
<td>Anna</td>
<td>Chloe</td>
</tr>
</tbody>
</table>

SoccerFan

<table>
<thead>
<tr>
<th>p</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anna</td>
<td>t</td>
</tr>
<tr>
<td>Bob</td>
<td>?</td>
</tr>
<tr>
<td>Chloe</td>
<td>?</td>
</tr>
</tbody>
</table>

OperaFan

<table>
<thead>
<tr>
<th>p</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anna</td>
<td>f</td>
</tr>
<tr>
<td>Bob</td>
<td>?</td>
</tr>
<tr>
<td>Chloe</td>
<td>t</td>
</tr>
</tbody>
</table>
Running Example

<table>
<thead>
<tr>
<th>Follows</th>
<th>SoccerFan</th>
<th>OperaFan</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>p</td>
<td>p</td>
</tr>
<tr>
<td>Anna</td>
<td>Anna</td>
<td>Anna</td>
</tr>
<tr>
<td>Bob</td>
<td>Bob</td>
<td>Bob</td>
</tr>
<tr>
<td>Anna</td>
<td>Chloe</td>
<td>Chloe</td>
</tr>
</tbody>
</table>

We wish to guess the missing values
We wish to guess the missing values; but no reason to prefer one guess to another.
### Running Example

<table>
<thead>
<tr>
<th>Follows</th>
<th>SoccerFan</th>
<th>OperaFan</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>p</td>
<td>p</td>
</tr>
<tr>
<td>Anna</td>
<td>Anna</td>
<td>Anna</td>
</tr>
<tr>
<td>Bob</td>
<td>Bob</td>
<td>Bob</td>
</tr>
<tr>
<td>Anna</td>
<td>Chloe</td>
<td>Chloe</td>
</tr>
<tr>
<td>Chloe</td>
<td>?</td>
<td>?</td>
</tr>
</tbody>
</table>

We wish to guess the missing values; but no reason to prefer one guess to another... yet
Soft Inference Rules

<table>
<thead>
<tr>
<th>Follows</th>
<th>SoccerFan</th>
<th>OperaFan</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>p</td>
<td>p</td>
</tr>
<tr>
<td>Anna</td>
<td>Anna</td>
<td>Anna</td>
</tr>
<tr>
<td>Bob</td>
<td>Bob</td>
<td>Bob</td>
</tr>
<tr>
<td>Anna</td>
<td>Chloe</td>
<td>Chloe</td>
</tr>
</tbody>
</table>

\[
\begin{align*}
\text{SoccerFan}(x) \land \text{Follows}(y, x) & \rightarrow \text{SoccerFan}(y) \\
\text{OperaFan}(x) \land \text{Follows}(y, x) & \rightarrow \text{OperaFan}(y) \\
\text{Soccer}(x) & \leftrightarrow \neg (\text{OperaFan}(x))
\end{align*}
\]
### Soft Inference Rules

<table>
<thead>
<tr>
<th>Follows</th>
<th>SoccerFan</th>
<th>OperaFan</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>p</td>
<td>p</td>
</tr>
<tr>
<td>Anna</td>
<td>Anna</td>
<td>Anna</td>
</tr>
<tr>
<td>Bob</td>
<td>Bob</td>
<td>Bob</td>
</tr>
<tr>
<td>Anna</td>
<td>Chloe</td>
<td>Chloe</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>p2</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob</td>
<td>t</td>
<td>f</td>
</tr>
<tr>
<td>Chloe</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>Chloe</td>
<td>?</td>
<td>t</td>
</tr>
</tbody>
</table>

\[
4: \ (\text{SoccerFan}(x) \land \text{Follows}(y, x)) \rightarrow \text{SoccerFan}(y) \\
2: \ (\text{OperaFan}(x) \land \text{Follows}(y, x)) \rightarrow \text{OperaFan}(y) \\
8: \ \text{Soccer}(x) \leftrightarrow \neg(\text{OperaFan}(x))
\]
Interpreting Soft Rules

- The semantics of such a database with soft rules is a probability space over the completions of the unknown values.
The semantics of such a database with soft rules is a probability space over the completions of the unknown values.

How softness and weighting translate into probabilities?

Various semantics in the literature, e.g., Probabilistic Datalog [Fuh00], ProbLog [RKT07], Probabilistic Soft Logic [BMG10], Probabilistic-Programming Datalog [BtCK+17]
The semantics of such a database with soft rules is a probability space over the completions of the unknown values.

How softness and weighting translate into probabilities?

- Various semantics in the literature, e.g., Probabilistic Datalog [Fuh00], ProbLog [RKT07], Probabilistic Soft Logic [BMG10], Probabilistic-Programming Datalog [BtCK+17]

- We will look at one such a translation: *Markov Logic Network (MLN)* [RD06]
We have a sequence $z = z_1, \ldots, z_n$ of (correlated) random variables, each $z_i$ taking values from a domain $\text{dom}(z_i)$.
We have a sequence $\mathbf{z} = z_1, \ldots, z_n$ of (correlated) random variables, each $z_i$ taking values from a domain $\text{dom}(z_i)$.

We consider the representation of a probability space over all possible assignments.
A factor over $z = z_1, \ldots, z_n$ is a function

$$\phi : \text{dom}(z_1) \times \cdots \times \text{dom}(z_n) \rightarrow [0, \infty)$$
A factor over \( z = z_1, \ldots, z_n \) is a function

\[
\phi : \text{dom}(z_1) \times \cdots \times \text{dom}(z_n) \rightarrow [0, \infty)
\]

We are typically interested in situations where each factor looks at only a small subset of the variables, e.g.:

\[
\phi(a) = \begin{cases} 
2 & \text{if } a_1 = 0 \\
5 & \text{if } a_1 = 1 \text{ and } a_7 = 0 \\
15 & \text{if } a_1 = 1 \text{ and } a_7 = 1 
\end{cases}
\]
A factor graph is a representation of a probability space over the assignments to \( z \)
Factor Graphs

- A factor graph is a representation of a probability space over the assignments to \( z \).
- Formally, a \textit{factor graph} for \( z \) is a sequence \( F = \phi_1, \ldots, \phi_m \) of factors over \( z \).
A factor graph is a representation of a probability space over the assignments to $z$

Formally, a factor graph for $z$ is a sequence $F = \phi_1, \ldots, \phi_m$ of factors over $z$

Semantics:

$$Pr(z = a) \overset{\text{def}}{=} \frac{1}{Z} \prod_{\phi_i \in F} \phi_i(a)$$
A factor graph is a representation of a probability space over the assignments to $z$

Formally, a *factor graph* for $z$ is a sequence $F = \phi_1, \ldots, \phi_m$ of factors over $z$

Semantics:

$$\Pr(z = a) \overset{\text{def}}{=} \frac{1}{Z} \prod_{\phi_i \in F} \phi_i(a)$$

$Z$ is a *normalization term*:

$$Z \overset{\text{def}}{=} \sum_{a} \prod_{\phi_i \in F} \phi_i(a)$$
Visually, it is convenient and conventional to represent a factor graph by a bipartite graph with:

- A node $v_f$ for each factor $f$
- A node $u_{z_i}$ for every random variable $z_i$ in $z$
- An edge between $v_f$ and $u_{z_i}$ whenever $z_i$ affects $f$
Example

Diagram showing the relationship between factors and individuals.

Factors:
- $f_1$, $f_2$, $f_3$, $f_4$, $f_5$, $f_6$, $f_7$, $f_8$, $f_9$

Individuals:
- Anna Soccer Fan
- Anna Opera Fan
- Bob Soccer Fan
- Bob Opera Fan
- Chloe Soccer Fan
- Chloe Opera Fan
Example: Naïve Bayes Classifier

\[
\Pr(y = c, x = a) = \Pr(y = c) \cdot \prod_{i=1}^{n} \Pr(x_i = a_i | y = c)
\]

\[\phi(c) \cdot \prod_{i=1}^{n} \phi_i(a_i, c)\]
Example: Naïve Bayes Classifier

\[
\Pr(y = c, \mathbf{x} = \mathbf{a}) = \Pr(y = c) \cdot \prod_{i=1}^{n} \Pr(x_i = a_i \mid y = c)
\]
Example: Naïve Bayes Classifier

\[
\Pr(y = c, \mathbf{x} = \mathbf{a}) = \Pr(y = c) \cdot \prod_{i=1}^{n} \Pr(x_i = a_i \mid y = c)
\]

\[
= \phi(c) \cdot \prod_{i=1}^{n} \phi_i(a_i, c)
\]
Example: Logistic Regression
Example: Logistic Regression

\[
\Pr(y = c, \mathbf{x} = \mathbf{a}) = \frac{e^{\beta_{c,0} + \sum \beta_{c,i} \cdot a_i}}{Z}
\]
Example: Hidden Markov Model
Example: Hidden Markov Model

\[
\Pr(y = c, x = a) = \Pr(y_1 = c_1) \cdot \Pr(x_1 = a_1 \mid y_1 = c_1) \\
\quad \times \Pr(y_2 = c_2 \mid y_1 = c_1) \cdot \Pr(x_2 = a_2 \mid y_2 = c_2) \\
\quad \times \cdots \\
\quad \times \Pr(y_n = c_n \mid y_{n-1} = c_{n-1}) \cdot \Pr(x_n = a_n \mid y_n = c_n)
\]
A Markov Logic Network (MLN) is a formalism for compactly encoding factor graphs using logic.
A *Markov Logic Network* (MLN) is a formalism for compactly encoding factor graphs using logic

- facts $\Rightarrow$ Boolean random variables
- rules $\Rightarrow$ factors
Some Applications of MLNs

- Entity resolution [SD06]
- Information extraction (extract structured data from text) [NRDS11]
- Robotics (scene analysis) [WD08]
- Social-network analysis
- Bioinformatics (protein interaction)
Setup

- We have a relational signature $\mathcal{R}$ and a finite domain $\mathcal{C}$ of attribute values
We have a relational signature $\mathcal{R}$ and a finite domain $C$ of attribute values

- For example, $\mathcal{R} = \{\text{Follows}/2, \text{SoccerFan}/1, \text{OperaFan}/1\}$ and $C = \{\text{Anna}, \text{Bob}, \text{Chloe}\}$
We have a relational signature $\mathcal{R}$ and a finite domain $C$ of attribute values

- For example, $\mathcal{R} = \{\text{Follows}/2, \text{SoccerFan}/1, \text{OperaFan}/1\}$ and $C = \{\text{Anna, Bob, Chloe}\}$
- For simplicity, same domain for all attributes
A \textit{ground fact} over $\mathcal{R}$ is a fact over $\mathcal{R}$ with values from $C$

- e.g., SoccerFan(Anna), Follows(Bob, Anna)
A ground fact over $R$ is a fact over $R$ with values from $C$
  - e.g., SoccerFan(Anna), Follows(Bob, Anna)
A Markov Logic Network (MLN) represents a probability space over all possible sets of ground facts.
A ground fact over $\mathcal{R}$ is a fact over $\mathcal{R}$ with values from $\mathcal{C}$
- e.g., SoccerFan(Anna), Follows(Bob,Anna)

A Markov Logic Network (MLN) represents a probability space over all possible sets of ground facts

We view a ground fact $f$ as a random Boolean variable $x_f$
- $x_f = \text{true}$ means that the fact $f$ is true (e.g., Anna is indeed a soccer fan)
A ground fact over $\mathcal{R}$ is a fact over $\mathcal{R}$ with values from $\mathcal{C}$
  - e.g., SoccerFan(Anna), Follows(Bob,Anna)

A Markov Logic Network (MLN) represents a probability space over all possible sets of ground facts

We view a ground fact $f$ as a random Boolean variable $x_f$
  - $x_f = \text{true}$ means that the fact $f$ is true (e.g., Anna is indeed a soccer fan)

An MLN is then a probability space over the $x_f$, assigning random true/false to each variable
Formal Definition

- An **MLN** over the signature $\mathcal{R}$ and domain $\mathcal{C}$ is a sequence $\langle w_1 : r_1, \ldots, w_k : r_k \rangle$ of weighted rules.
An **MLN** over the signature $\mathcal{R}$ and domain $\mathcal{C}$ is a sequence $\langle w_1 : r_1, \ldots, w_k : r_k \rangle$ of weighted rules:

- Each $r_i$ is a propositional formula with free variables.
An **MLN** over the signature \( \mathcal{R} \) and domain \( \mathcal{C} \) is a sequence \( \langle w_1 : r_1, \ldots, w_k : r_k \rangle \) of weighted rules

- Each \( r_i \) is a propositional formula with free variables
- Each \( w_i \) is a nonnegative number
Formal Definition

- An MLN over the signature \( \mathcal{R} \) and domain \( \mathcal{C} \) is a sequence \( \langle w_1 : r_1, \ldots, w_k : r_k \rangle \) of weighted rules
  - Each \( r_i \) is a propositional formula with free variables
  - Each \( w_i \) is a nonnegative number
  - Zeros are important for encoding hard constraints (must hold in every world)
Formal Definition

- An **MLN** over the signature \( \mathcal{R} \) and domain \( \mathcal{C} \) is a sequence \( \langle w_1 : r_1, \ldots, w_k : r_k \rangle \) of weighted rules
  - Each \( r_i \) is a propositional formula with free variables
  - Each \( w_i \) is a nonnegative number
  - Zeros are important for encoding hard constraints (must hold in every world)
- A **grounding** of a rule \( r \) is obtained by replacing the free variables with constants
Formal Definition

- An **MLN** over the signature $\mathcal{R}$ and domain $\mathcal{C}$ is a sequence $\langle w_1 : r_1, \ldots, w_k : r_k \rangle$ of weighted rules
  - Each $r_i$ is a propositional formula with free variables
  - Each $w_i$ is a nonnegative number
  - Zeros are important for encoding hard constraints (must hold in every world)

- A **grounding** of a rule $r$ is obtained by replacing the free variables with constants; for example:

\[
\left( \text{OperaFan}(x) \land \text{Follows}(y, x) \right) \rightarrow \text{OperaFan}(y)
\]

\[
\Downarrow
\]

\[
\left( \text{OperaFan}(\text{Anna}) \land \text{Follows}(\text{Bob, Anna}) \right) \rightarrow \text{OperaFan}(\text{Bob})
\]
Formal Definition (continued)

- The MLN $\langle w_1 : r_1, \ldots, w_k : r_k \rangle$ represents a factor graph over the $x_f$
The MLN $\langle w_1 : r_1, \ldots, w_k : r_k \rangle$ represents a factor graph over the $x_f$.

A possible assignment $a$ to the sequence $x$ of variables represents an instance over $\mathcal{R}$; we denote it by $I_a$.

- $I_a$ consists of all the facts $f$ such that $a_f = \text{true}$
The MLN $\langle w_1 : r_1, \ldots, w_k : r_k \rangle$ represents a factor graph over the $x_f$

A possible assignment $a$ to the sequence $x$ of variables represents an instance over $\mathcal{R}$; we denote it by $I_a$

- $I_a$ consists of all the facts $f$ such that $a_f = \text{true}$

Each grounding $g$ of $r_i$ provides a factor, denoted $\phi_{g,r_i}$
The MLN $\langle w_1 : r_1, \ldots, w_k : r_k \rangle$ represents a factor graph over the $x_f$.

A possible assignment $a$ to the sequence $x$ of variables represents an instance over $\mathcal{R}$; we denote it by $I_a$.

- $I_a$ consists of all the facts $f$ such that $a_f = \text{true}$.

Each grounding $g$ of $r_i$ provides a factor, denoted $\phi_{g,r_i}$

$$\phi_{g,r_i}(a) = \begin{cases} w_i & \text{if } g(r_i) \text{ is true in the assignment } I_a; \\ 1 & \text{otherwise.} \end{cases}$$
Formal Definition (continued)

- The MLN $\langle w_1 : r_1, \ldots, w_k : r_k \rangle$ represents a factor graph over the $x_f$
- A possible assignment $a$ to the sequence $x$ of variables represents an instance over $\mathcal{R}$; we denote it by $I_a$
  - $I_a$ consists of all the facts $f$ such that $a_f = \text{true}$
- Each grounding $g$ of $r_i$ provides a factor, denoted $\phi_{g,r_i}$
  $$\phi_{g,r_i}(a) = \begin{cases} 
  w_i & \text{if } g(r_i) \text{ is true in the assignment } I_a; \\
  1 & \text{otherwise.}
\end{cases}$$
- What are the random variables that connect to $\phi_{g,r_i}$?
The MLN \( \langle w_1 : r_1, \ldots, w_k : r_k \rangle \) represents a factor graph over the \( x_f \).

A possible assignment \( a \) to the sequence \( x \) of variables represents an instance over \( \mathcal{R} \); we denote it by \( I_a \).

- \( I_a \) consists of all the facts \( f \) such that \( a_f = \text{true} \).

Each grounding \( g \) of \( r_i \) provides a factor, denoted \( \phi_{g,r_i} \).

\[
\phi_{g,r_i}(a) = \begin{cases} 
w_i & \text{if } g(r_i) \text{ is true in the assignment } I_a; \\
1 & \text{otherwise.}
\end{cases}
\]

What are the random variables that connect to \( \phi_{g,r_i} \)?

Following Poole [Poo03], an MLN rule is called a parametric factor, or parfactor for short.

- Arise in other models, e.g., *Probabilistic Soft Logic* [BMG10].
Example Revisited

<table>
<thead>
<tr>
<th>Follows</th>
<th>SoccerFan</th>
<th>OperaFan</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>p</td>
<td>p</td>
</tr>
<tr>
<td>Anna</td>
<td>Anna</td>
<td>Anna</td>
</tr>
<tr>
<td>Bob</td>
<td>Bob</td>
<td>Bob</td>
</tr>
<tr>
<td>Anna</td>
<td>Chloe</td>
<td>Chloe</td>
</tr>
</tbody>
</table>

4: \((\text{SoccerFan}(x) \land \text{Follows}(y, x)) \rightarrow \text{SoccerFan}(y)\)

2: \((\text{OperaFan}(x) \land \text{Follows}(y, x)) \rightarrow \text{OperaFan}(y)\)

8: \(\text{Soccer}(x) \iff \neg(\text{OperaFan}(x))\)
**What are the factors here?**

<table>
<thead>
<tr>
<th></th>
<th>Follows</th>
<th>SoccerFan</th>
<th>OperaFan</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>p2</td>
<td>p</td>
<td></td>
</tr>
<tr>
<td>Anna</td>
<td>Bob</td>
<td>Anna</td>
<td>Anna</td>
</tr>
<tr>
<td>Bob</td>
<td>Chloe</td>
<td>Bob</td>
<td>Bob</td>
</tr>
<tr>
<td>Anna</td>
<td>Chloe</td>
<td>Chloe</td>
<td>Chloe</td>
</tr>
</tbody>
</table>

- **Follows**:
  - Anna → Bob
  - Bob → Chloe
  - Anna → Chloe

- **SoccerFan**:
  - Anna → t
  - Bob → t
  - Chloe → f

- **OperaFan**:
  - Anna → f
  - Bob → t
  - Chloe → t

\[ 4 : (\text{SoccerFan}(x) \land \text{Follows}(y, x)) \rightarrow \text{SoccerFan}(y) \]

\[ 2 : (\text{OperaFan}(x) \land \text{Follows}(y, x)) \rightarrow \text{OperaFan}(y) \]

\[ 8 : \text{Soccer}(x) \leftrightarrow \neg (\text{OperaFan}(x)) \]
Core MLN Tasks

- MLNs entail several computational tasks
Core MLN Tasks

- MLNs entail several computational tasks
  - Estimating marginal probability
    - \( z = (x, y, v) \), and given \( x \) and \( y \), compute \( \Pr(y | x) \)
Core MLN Tasks

- MLNs entail several computational tasks
  - Estimating marginal probability
    - $z = (x, y, v)$, and given $x$ and $y$, compute $Pr(y | x)$
  - Maximum A-Posteriori (MAP) inference
    - $z = (x, y)$, and given $x$, compute $\text{argmax}_y Pr(y | x)$
Core MLN Tasks

- MLNs entail several computational tasks
  - Estimating marginal probability
    - $z = (x, y, v)$, and given $x$ and $y$, compute $\Pr(y \mid x)$
  - Maximum A-Posteriori (MAP) inference
    - $z = (x, y)$, and given $x$, compute $\arg\max_y \Pr(y \mid x)$
  - Weight learning: given examples of domains and worlds, find the most likely rule weights to produce the worlds (fitting)
Core MLN Tasks

- MLNs entail several computational tasks
  - Estimating marginal probability
    - \( z = (x, y, v) \), and given \( x \) and \( y \), compute \( \Pr(y \mid x) \)
  - Maximum A-Posteriori (MAP) inference
    - \( z = (x, y) \), and given \( x \), compute \( \text{argmax}_y \Pr(y \mid x) \)
  - Weight learning: given examples of domains and worlds, find the most likely rule weights to produce the worlds (fitting)
  - Structure learning: given examples of domains and worlds, find “good” rules and weights that are likely to produce the worlds
Core MLN Tasks

- MLNs entail several computational tasks
  - Estimating marginal probability
    - \( z = (x, y, v) \), and given \( x \) and \( y \), compute \( \Pr(y \mid x) \)
  - Maximum A-Posteriori (MAP) inference
    - \( z = (x, y) \), and given \( x \), compute \( \text{argmax}_y \Pr(y \mid x) \)
  - Weight learning: given examples of domains and worlds, find the most likely rule weights to produce the worlds (fitting)
  - Structure learning: given examples of domains and worlds, find “good” rules and weights that are likely to produce the worlds
- These tasks are usually intractable, and common techniques include heuristic local search (MaxWalkSat), sampling (MCMC), gradient-based optimization, message passing, and symmetry-based simplification (lifted inference)
Jha and Suciu [JS12] show how to translate MLNs to tuple-independent databases.
Jha and Suciu [JS12] show how to translate MLNs to tuple-independent databases

I will show a simplified version of that translation on our example
Translation Example (1)

Rule $r(x, y)$:

$$4: \ (\text{SoccerFan}(x) \land \text{Follows}(y, x)) \rightarrow \text{SoccerFan}(y)$$
Translation Example (1)

Rule \( r(x, y) \):

\[ 4: \left( \text{SoccerFan}(x) \land \text{Follows}(y, x) \right) \rightarrow \text{SoccerFan}(y) \]
Translation Example (1)

Rule $r(x, y)$:

4: \[(\text{SoccerFan}(x) \land \text{Follows}(y, x)) \rightarrow \text{SoccerFan}(y)\]

↓

1. Initialize an empty TID $I$
Translation Example (1)

Rule $r(x, y)$:

4: $(\text{SoccerFan}(x) \land \text{Follows}(y, x)) \rightarrow \text{SoccerFan}(y)$

\[\downarrow\]

1. Initialize an empty TID $I$
2. Insert into $I$ each ground fact with probability $1/2$
   - SoccerFan(Anna):0.5, Follows(Bob,Anna):0.5,…
Translation Example (1)

Rule $r(x, y)$:

$$4: \ (\text{SoccerFan}(x) \land \text{Follows}(y, x)) \rightarrow \text{SoccerFan}(y)$$

1. Initialize an empty TID $I$
2. Insert into $I$ each ground fact with probability $1/2$
   - SoccerFan(Anna):0.5, Follows(Bob,Anna):0.5, ...
3. For each grounding $r(a, b)$ of $r(x, y)$ add a new fact $R(a, b)$ with probability $4/5$
Translation Example (1)

Rule $r(x, y)$:

$4: \left( \text{SoccerFan}(x) \land \text{Follows}(y, x) \right) \rightarrow \text{SoccerFan}(y)$

1. Initialize an empty TID $I$
2. Insert into $I$ each ground fact with probability $1/2$
   - SoccerFan(Anna):0.5, Follows(Bob,Anna):0.5,…
3. For each grounding $r(a, b)$ of $r(x, y)$ add a new fact $R(a, b)$ with probability $4/5$

Define:

$$Q_r() := \exists_{x,y} \left[ (R(x, y) \land \neg r(x, y)) \lor (\neg R(x, y) \land r(x, y)) \right]$$
Claim: Our MLN defines the probability distribution \([I]\), restricted to the MLN facts and conditioned on \(\neg Q_r()\).
Translation Example (2)

- Claim: Our MLN defines the probability distribution \([I]\), restricted to the MLN facts and conditioned on \(\neg Q_r()\)
- Proof idea: view a possible world of \(I\) as composing of two components: MLN facts \((W)\) and \(R\)-facts \((W_R)\)
  - Note: every \(W\) has the same probability in \(I\)
Translation Example (2)

- Claim: Our MLN defines the probability distribution $[I]$, restricted to the MLN facts and conditioned on $\neg Q_r()$
- Proof idea: view a possible world of $I$ as composing of two components: MLN facts ($W$) and $R$-facts ($W_R$)
  - Note: every $W$ has the same probability in $I$
- By using Bayes rule we get:

$$\Pr(W \mid \neg Q_r)$$
-- Claim: Our MLN defines the probability distribution $[I]$, restricted to the MLN facts and conditioned on $\neg Q_r()$

-- Proof idea: view a possible world of $I$ as composing of two components: MLN facts ($W$) and $R$-facts ($W_R$)
  -- Note: every $W$ has the same probability in $I$

-- By using Bayes rule we get:

$$\Pr(W \mid \neg Q_r) = \frac{\Pr(\neg Q_r \mid W) \times \Pr(W)}{\Pr(\neg Q_r)}$$
Claim: Our MLN defines the probability distribution $[I]$, restricted to the MLN facts and conditioned on $\neg Q_r()$

Proof idea: view a possible world of $I$ as composing of two components: MLN facts ($W$) and $R$-facts ($W_R$)

Note: every $W$ has the same probability in $I$

By using Bayes rule we get:

$$\Pr(W \mid \neg Q_r) = \frac{\Pr(\neg Q_r \mid W) \times \Pr(W)}{\Pr(\neg Q_r)} \sim \Pr(\neg Q_r \mid W)$$
Let us look at $\Pr(\neg Q_r \mid W)$
Translation Example (3)

- Let us look at $\Pr(\neg Q_r \mid W)$
- This is the probability that every $R(a, b)$ exists if and only if $r(a, b)$ holds
Translation Example (3)

- Let us look at $\Pr(\neg Q_r \mid W)$
- This is the probability that every $R(a, b)$ exists if and only if $r(a, b)$ holds
- Hence:

$$\Pr(\neg Q_r \mid W)$$
Translation Example (3)

- Let us look at $\Pr(\neg Q_r \mid W)$
- This is the probability that every $R(a, b)$ exists if and only if $r(a, b)$ holds
- Hence:

\[
\Pr(\neg Q_r \mid W) = \Pr(\bigwedge_{W|=r(a,b)} R(a, b) \land \bigwedge_{W|=\neg r(a,b)} \neg R(a, b) \mid W)
\]
Translation Example (3)

- Let us look at $\Pr(\neg Q_r \mid W)$
- This is the probability that every $R(a, b)$ exists if and only if $r(a, b)$ holds
- Hence:

$$\Pr(\neg Q_r \mid W) = \Pr(\bigwedge_{W \models r(a,b)} R(a, b) \land \bigwedge_{W \models \neg r(a,b)} \neg R(a, b) \mid W)$$

$$= \left( \prod_{W \models r(a,b)} \frac{4}{5} \right) \times \left( \prod_{W \models \neg r(a,b)} \frac{1}{5} \right)$$
Let us look at \( \Pr(\neg Q_r \mid W) \)

This is the probability that every \( R(a, b) \) exists if and only if \( r(a, b) \) holds

Hence:

\[
\Pr(\neg Q_r \mid W) = \Pr(\bigwedge_{W \models r(a, b)} R(a, b) \bigwedge_{W \models \neg r(a, b)} \neg R(a, b) \mid W)
\]

\[
= \left( \prod_{W \models r(a, b)} \frac{4}{5} \right) \times \left( \prod_{W \models \neg r(a, b)} \frac{1}{5} \right) \sim \prod_{W \models r(a, b)} 4 \prod_{W \models \neg r(a, b)} 1
\]
Translation Example (3)

- Let us look at $\Pr(\neg Q_r | W)$
- This is the probability that every $R(a, b)$ exists if and only if $r(a, b)$ holds
- Hence:

$$
\Pr(\neg Q_r | W) = \Pr(\bigwedge_{W \models r(a, b)} R(a, b) \land \bigwedge_{W \models \neg r(a, b)} \neg R(a, b) | W)
= \left( \prod_{W \models r(a, b)} \frac{4}{5} \right) \times \left( \prod_{W \models \neg r(a, b)} \frac{1}{5} \right) \sim \prod_{W \models r(a, b)} 4 \prod_{W \models \neg r(a, b)} 1
= \prod_{W \models r(a, b)} 4
$$
The probability of a query $Q$ over the MLN is then:

$$\Pr(I \vDash Q | I \vDash \neg Q_r)$$
The probability of a query $Q$ over the MLN is then:

$$\Pr(I \models Q \mid I \models \neg Q_r) = \frac{\Pr(I \models Q \land \neg Q_r)}{\Pr(I \models \neg Q_r)}$$
The probability of a query $Q$ over the MLN is then:

$$\Pr(I \models Q \mid I \models \neg Q_r) = \frac{\Pr(I \models Q \land \neg Q_r)}{\Pr(I \models \neg Q_r)}$$

$$= \frac{\Pr(I \models Q) - \Pr(I \models Q \land Q_r)}{1 - \Pr(I \models Q_r)}$$
# Table of Contents

1. Introduction
2. Markov Logic Networks
3. Probabilistic XML
XML Example

```xml
<aerial-photo>
  <region>
    <neighborhood>
      <house size="s">
        <vehicle type="private">
          <neighborhood>
            <factory>
              <facility/>
              <heliport/>
            </factory>
          </neighborhood>
        </vehicle>
      </house>
    </neighborhood>
  </region>
</aerial-photo>
```

Diagram:
- aerial photo
  - region
    - neighborhood
      - house (size: s)
        - vehicle (type: private)
          - neighborhood
            - factory
              - facility
              - heliport

- neighborhood
- factory
- house (size: s)
- vehicle (type: private)
- facility
- heliport
Probabilistic XML (p-Document) Example

- aerial photo
- region
  - neighborhood
    - house
      - size
        - s
    - house
      - size
        - m
  - vehicle
    - type
    - track
    - private
  - factory
    - facility
      - parking lot
      - heliport
The p-Document Model

- A *d-document* is a tree with two types of nodes
A \textit{d-document} is a tree with two types of nodes

- An \textit{ordinary} node is an XML node with a label (or text)
The p-Document Model

- A **d-document** is a tree with two types of nodes
  - An **ordinary** node is an XML node with a label (or text)
  - A **distributional** node is an encoding of a probabilistic choice of a random set of children
The p-Document Model

- A *d-document* is a tree with two types of nodes
  - An *ordinary* node is an XML node with a label (or text)
  - A *distributional* node is an encoding of a probabilistic choice of a random set of children
- The *root* of a p-document is always an ordinary node
The p-Document Model

- A \textit{d-document} is a tree with two types of nodes
  - An \textit{ordinary} node is an XML node with a label (or text)
  - A \textit{distributional} node is an encoding of a probabilistic choice of a random set of children
- The \textit{root} of a p-document is always an ordinary node
- There are various types and representations of distributional nodes:
The p-Document Model

- A **d-document** is a tree with two types of nodes
  - An **ordinary** node is an XML node with a label (or text)
  - A **distributional** node is an encoding of a probabilistic choice of a random set of children
- The **root** of a p-document is always an ordinary node
- There are various types and representations of distributional nodes:
  - **Independent** choice of children, each child has an associated probability
The p-Document Model

- A d-document is a tree with two types of nodes
  - An ordinary node is an XML node with a label (or text)
  - A distributional node is an encoding of a probabilistic choice of a random set of children
- The root of a p-document is always an ordinary node
- There are various types and representations of distributional nodes:
  - Independent choice of children, each child has an associated probability
  - Mutually-exclusive choice, children probabilities sum up to (at most) one
The p-Document Model

- A *d-document* is a tree with two types of nodes
  - An *ordinary* node is an XML node with a label (or text)
  - A *distributional* node is an encoding of a probabilistic choice of a random set of children
- The *root* of a p-document is always an ordinary node
- There are various types and representations of distributional nodes:
  - Independent choice of children, each child has an associated probability
  - Mutually-exclusive choice, children probabilities sum up to (at most) one
  - Shared random variables (correlations, similar to pc-tables)
Sampling Example
A p-document represents a probability space over ordinary XML documents
Semantics of a p-Document

- A p-document represents a probability space over ordinary XML documents
- To defined this probability space, we need to explain how a random document is being generated/sampled
A p-document represents a probability space over ordinary XML documents.

To defined this probability space, we need to explain how a random document is being generated/sampled.

Top-down process; for each distributional node $v$:

1. Randomly select a subset of $v$’s children (according to the local distribution of $v$)
2. Throw away unchosen children (and their subtrees)
3. Eliminate $v$ by connecting the chosen children directly to $v$’s parent
Some Studied Problems

- *Data integration* with probabilistic XML [vKdKA05]
Some Studied Problems

- *Data integration* with probabilistic XML [vKdKA05]
- Evaluating *XPath queries* over p-documents [KKS09, ACK+11]
Some Studied Problems

- *Data integration* with probabilistic XML [vKdKA05]
- Evaluating *XPath queries* over p-documents [KKS09, ACK+11]
- Computing the probability of *DTD compatibility* [CKS09]
  - More generally, the probability of acceptance by a *tree automaton*
Some Studied Problems

- **Data integration** with probabilistic XML [vKdKA05]
- Evaluating *XPath queries* over p-documents [KKS09, ACK+11]
- Computing the probability of *DTD compatibility* [CKS09]
  - More generally, the probability of acceptance by a *tree automaton*
- Probabilistic XML defined by *probabilistic context-free grammars* (PCFGs) [BKOS10, CK10]
Some Studied Problems

- **Data integration** with probabilistic XML [vKdKA05]
- Evaluating *XPath queries* over p-documents [KKS09, ACK+11]
- Computing the probability of *DTD compatibility* [CKS09]
  - More generally, the probability of acceptance by a *tree automaton*
- Probabilistic XML defined by *probabilistic context-free grammars* (PCFGs) [BKOS10, CK10]
- *Keyword search* over probabilistic XML [LLZW11]


References II


References III


End of lecture 12

More on Probabilistic Databases