First-Order
Knowledge Compilation

Guy Van den Broeck

Dagstuhl
Sept 18, 2017
Overview

1. Propositional Refresher
2. Primer: A First-Order Tractable Language
3. Probabilistic Databases
4. Symmetric First-Order Model Counting
5. Lots of Pointers
Overview

1. Propositional Refresher
2. Primer: A First-Order Tractable Language
3. Probabilistic Databases
4. Symmetric First-Order Model Counting
5. Lots of Pointers
Negation Normal Form

\[ \Delta = (\text{sun} \land \text{rain} \Rightarrow \text{rainbow}) \]
Decomposable NNF

[Darwiche 2002]
Deterministic NNF

[Darwiche 2002]
Model Counting

- Model = solution to a propositional logic formula $\Delta$
- Model counting = $\#\text{SAT}$

$\Delta = (\text{Rain} \Rightarrow \text{Cloudy})$

<table>
<thead>
<tr>
<th>Rain</th>
<th>Cloudy</th>
<th>Model?</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>Yes</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>No</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>Yes</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>Yes</td>
</tr>
</tbody>
</table>

$\#\text{SAT} = 3$
Model Counting

- Model = solution to a propositional logic formula $\Delta$
- Model counting = \#SAT

$\Delta = (\text{Rain} \Rightarrow \text{Cloudy})$

<table>
<thead>
<tr>
<th>Rain</th>
<th>Cloudy</th>
<th>Model?</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>Yes</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>No</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>Yes</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>Yes</td>
</tr>
</tbody>
</table>

$\#\text{SAT} = 3$

[Valiant] \#P-hard, even for 2CNF
Deterministic Decomposable NNF

Model Counting?

[Darwiche 2002]
Deterministic Decomposable NNF

Model Counting

[Darwiche 2002]
Weighted Model Count

- Weights for assignments to variables
- Model weight = product of variable weights

\[ \Delta = (\text{Rain} \Rightarrow \text{Cloudy}) \]

<table>
<thead>
<tr>
<th>Rain</th>
<th>Cloudy</th>
<th>Model?</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>Yes</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>No</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>Yes</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Weighted Model Count

- Weights for assignments to variables
- Model weight = product of variable weights

\[ \Delta = (\text{Rain} \Rightarrow \text{Cloudy}) \]

<table>
<thead>
<tr>
<th>Rain</th>
<th>Cloudy</th>
<th>Model?</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>Yes</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>No</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>Yes</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Weighted Model Count

- Weights for assignments to variables
- Model weight = product of variable weights

\[ \Delta = (\text{Rain} \Rightarrow \text{Cloudy}) \]

<table>
<thead>
<tr>
<th>Rain</th>
<th>Clouidy</th>
<th>Model?</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>Yes</td>
<td>1 \times 3 = 3</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>No</td>
<td>1 \times 0 = 0</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>Yes</td>
<td>2 \times 3 = 6</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>Yes</td>
<td>2 \times 5 = 10</td>
</tr>
</tbody>
</table>

\[ \text{WMC} = 19 \]
Deterministic Decomposable NNF

Weighted Model Counting

[Darwiche 2002]
Assembly language for probabilistic reasoning

- Bayesian networks
- Factor graphs
- Probabilistic logic programs
- Relational Bayesian networks
- Probabilistic databases
- Markov Logic
- Weighted Model Counting

Probability of a Sentence

• Special case of WMC
• Weights are probabilities: \( w(R) + w(\neg R) = 1 \)

\[ \Delta = (\text{Rain} \Rightarrow \text{Cloudy}) \]

<table>
<thead>
<tr>
<th>Rain</th>
<th>Cloudy</th>
<th>Model?</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>Yes</td>
<td>.8 * .5 = .4</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>No</td>
<td>0</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>Yes</td>
<td>.2 * 5 = .1</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>Yes</td>
<td>.2 * 5 = .1</td>
</tr>
</tbody>
</table>

• Simplifies some details (smoothing)
Probability of a Sentence

- Special case of WMC
- Weights are probabilities: \( w(R) + w(\neg R) = 1 \)

\[ \Delta = (\text{Rain} \Rightarrow \text{Cloudy}) \]

<table>
<thead>
<tr>
<th>Rain</th>
<th>Cloudy</th>
<th>Model?</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>Yes</td>
<td>(.8 \times .5 = .4)</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>No</td>
<td>0</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>Yes</td>
<td>(.2 \times 5 = .1)</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>Yes</td>
<td>(.2 \times 5 = .1)</td>
</tr>
</tbody>
</table>

\[ P(\Delta) = 0.6 \]

- Simplifies some details (smoothing)
Overview

1. Propositional Refresher

2. Primer: A First-Order Tractable Language

3. Probabilistic Databases

4. Symmetric First-Order Model Counting

5. Lots of Pointers
First-Order NNF

\( \forall X, X \in \text{People} : \text{belgian}(X) \Rightarrow \text{likes}(X, \text{chocolate}) \)
First-Order Decomposability

\[ \forall X \in \text{People} \]

[Van den Broeck 2013]
First-Order Decomposability

\[ \forall X \in \text{People} \]

\[ \text{belgian}(X) \lor \text{likes}(X, chocolate) \lor \neg \text{belgian}(X) \]

[Van den Broeck 2013]
First-Order Determinism

\[ \forall X \ (X \in \text{People}) \]

Deterministic

\[ \neg \text{belgian}(X) \]

\[ \text{likes}(X, \text{chocolate}) \]

[Van den Broeck 2013]
Probability of Sentence (WMC)

\[ \forall X \in \text{People} \]

\[ \land \]

\[ \lor \]

\[ \land \]

\[ \text{belgian}(X) \]

\[ \text{likes}(X, \text{chocolate}) \]

\[ \neg \text{belgian}(X) \]

[Van den Broeck 2013]
Probability of Sentence (WMC)

For $X = \text{guy}$: $\cdot92$

[Van den Broeck 2013]
Probability of Sentence (WMC)

For $X = \text{guy}$: $0.92$
$X = \text{mary}$: $0.97$

[Van den Broeck 2013]
Probability of Sentence (WMC)

For all people:
\[ 0.92 \times 0.97 = 0.89 \]

[Van den Broeck 2013]
Evaluate Probability on FO Circuit*

* Also non-NNF to simplify examples. Some rules redundant given others.
Evaluate Probability on FO Circuit*

\[ P(\neg Q) = 1 - P(Q) \]

Negation

* Also non-NNF to simplify examples. Some rules redundant given others.
Evaluate Probability on FO Circuit*

\[ P(\neg Q) = 1 - P(Q) \]

\[ P(Q_1 \land Q_2) = P(Q_1) \cdot P(Q_2) \]
\[ P(Q_1 \lor Q_2) = 1 - (1 - P(Q_1)) \cdot (1 - P(Q_2)) \]

Negation

Decomposable \(\land, \lor\)

*Also non-NNF to simplify examples. Some rules redundant given others.
Evaluate Probability on FO Circuit*

\[ P(\neg Q) = 1 - P(Q) \]

\[ P(Q_1 \land Q_2) = P(Q_1) \cdot P(Q_2) \]
\[ P(Q_1 \lor Q_2) = 1 - (1 - P(Q_1)) \cdot (1 - P(Q_2)) \]

\[ P(\forall z \ Q) = \prod_{A \in \text{Domain}} P(Q[A/z]) \]
\[ P(\exists z \ Q) = 1 - \prod_{A \in \text{Domain}} (1 - P(Q[A/z])) \]

Negation

Decomposable \land, \lor

Decomposable \forall, \exists

* Also non-NNF to simplify examples. Some rules redundant given others.
Evaluate Probability on FO Circuit*

\[ P(\neg Q) = 1 - P(Q) \]

\[ P(Q_1 \land Q_2) = P(Q_1) \cdot P(Q_2) \]
\[ P(Q_1 \lor Q_2) = 1 - (1 - P(Q_1)) \cdot (1 - P(Q_2)) \]

\[ P(\forall z \ Q) = \prod_{A \in \text{Domain}} P(Q[A/z]) \]
\[ P(\exists z \ Q) = 1 - \prod_{A \in \text{Domain}} (1 - P(Q[A/z])) \]

\[ P(Q_1 \land Q_2) = P(Q_1) + P(Q_2) - 1 \]
\[ P(Q_1 \lor Q_2) = P(Q_1) + P(Q_2) \]

Negation

Decomposable \( \land, \lor \)

Decomposable \( \forall, \exists \)

Deterministic \( \land, \lor \)

* Also non-NNF to simplify examples. Some rules redundant given others.
Evaluate Probability on FO Circuit*

\[ P(\neg Q) = 1 - P(Q) \]

\[ P(Q_1 \land Q_2) = P(Q_1) P(Q_2) \]
\[ P(Q_1 \lor Q_2) = 1 - (1 - P(Q_1)) (1 - P(Q_2)) \]

\[ P(\forall z Q) = \prod_{A \in \text{Domain}} P(Q[A/z]) \]
\[ P(\exists z Q) = 1 - \prod_{A \in \text{Domain}} (1 - P(Q[A/z])) \]

\[ P(Q_1 \land Q_2) = P(Q_1) + P(Q_2) - 1 \]
\[ P(Q_1 \lor Q_2) = P(Q_1) + P(Q_2) \]

\[ P(\forall z Q) = 1 - \sum_{A \in \text{Domain}} 1 - P(Q[A/z]) \]
\[ P(\exists z Q) = \sum_{A \in \text{Domain}} P(Q[A/z]) \]

Negation

Decomposable \(\land, \lor\)

Decomposable \(\forall, \exists\)

Deterministic \(\land, \lor\)

Deterministic \(\forall, \exists\)

* Also non-NNF to simplify examples. Some rules redundant given others.
Limitations

\[ H_0 = \forall x \forall y \text{ Smoker}(x) \lor \text{ Friend}(x,y) \lor \text{ Jogger}(y) \]

The decomposable \( \forall \)-rule:

\[ P(\forall z \ Q) = \prod_{A \in \text{Domain}} P(Q[A/z]) \]
Limitations

\( H_0 = \forall x \forall y \text{Smoker}(x) \lor \text{Friend}(x,y) \lor \text{Jogger}(y) \)

The decomposable \( \forall \)-rule: \[ P(\forall z \ Q) = \prod_{A \in \text{Domain}} P(Q[A/z]) \] … does not apply:

\( H_0[Alice/x] \) and \( H_0[Bob/x] \) are dependent:

\( \forall y \ (\text{Smoker}(Alice) \lor \text{Friend(Alice,y)} \lor \text{Jogger(y)}) \)

\( \forall y \ (\text{Smoker}(Bob) \lor \text{Friend(Bob,y)} \lor \text{Jogger(y)}) \)

[Suciu’11]
Limitations

\( H_0 = \forall x \forall y \text{ Smoker}(x) \lor \text{ Friend}(x,y) \lor \text{ Jogger}(y) \)

The decomposable \( \forall \)-rule: \( P(\forall z \ Q) = \prod_{A \in \text{Domain}} P(Q[A/z]) \)

... does not apply:

\( H_0[Alice/x] \) and \( H_0[Bob/x] \) are dependent:

\[ \forall y \ (\text{Smoker}(Alice) \lor \text{Friend}(Alice,y) \lor \text{Jogger}(y)) \]

\[ \forall y \ (\text{Smoker}(Bob) \lor \text{Friend}(Bob,y) \lor \text{Jogger}(y)) \]

Is this FO circuit language not powerful enough?

[Suciu’11]
Background: Positive Partitioned 2CNF

A PP2CNF is:

\[ F = \bigwedge_{(i,j) \in E} (x_i \lor y_j) \]

where \( E = \) the edge set of a bipartite graph

\[ F = (x_1 \lor y_1) \land (x_2 \lor y_1) \land (x_2 \lor y_3) \land (x_1 \lor y_3) \land (x_2 \lor y_2) \]
Background:
Positive Partitioned 2CNF

A PP2CNF is:
\[
F = \bigwedge_{(i,j) \in E} (x_i \lor y_j)
\]
where \(E\) = the edge set of a bipartite graph

\[
F = (x_1 \lor y_1) \land (x_2 \lor y_1) \land (x_2 \lor y_3) \\
\land (x_1 \lor y_3) \land (x_2 \lor y_2)
\]

**Theorem:** \(#\text{PP2CNF}\) is \(#\text{P}\)-hard [Provan’83]
Our Problematic Clause

\[ H_0 = \forall x \forall y \text{ Smoker}(x) \lor \text{ Friend}(x,y) \lor \text{ Jogger}(y) \]
**Our Problematic Clause**

\[
H_0 = \forall x \forall y \text{ Smoker}(x) \lor \text{ Friend}(x,y) \lor \text{ Jogger}(y)
\]

**Theorem.** Computing \( P(H_0) \) is \( \#P \)-hard in the size of weight function \( w(.) \) (i.e., the number of people)

[Dalvi&S.’04]
Our Problematic Clause

\[ H_0 = \forall x \forall y \text{Smoker}(x) \lor \text{Friend}(x,y) \lor \text{Jogger}(y) \]

**Theorem.** Computing \( \mathbb{P}(H_0) \) is \( \#P \)-hard in the size of weight function \( w(.) \) (i.e., the number of people)

[Dalvi&S.'04]

**Proof:** PP2CNF: 

\[ F = (X_{i1} \lor Y_{j1}) \land (X_{i2} \lor Y_{j2}) \land \ldots \] reduce \#F to computing \( \mathbb{P}(H_0) \)

By example:
Our Problematic Clause

\[ H_0 = \forall x \forall y \text{ Smoker}(x) \lor \text{Friend}(x,y) \lor \text{Jogger}(y) \]

**Theorem.** Computing \( P(H_0) \) is \#P-hard in the size of weight function \( w(.) \) (i.e., the number of people)

[Dalvi&S.’04]

**Proof:** PP2CNF: \( F = (X_{i1} \lor Y_{j1}) \land (X_{i2} \lor Y_{j2}) \land \ldots \) reduce \( \#F \) to computing \( P(H_0) \)

By example:

\[ F = (X_1 \lor Y_1) \land (X_1 \lor Y_2) \land (X_2 \lor Y_2) \]
Our Problematic Clause

\[ H_0 = \forall x \forall y \text{ Smoker}(x) \lor \text{ Friend}(x,y) \lor \text{ Jogger}(y) \]

**Theorem.** Computing \( P(H_0) \) is \(#P\)-hard in the size of weight function \( w(.) \) (i.e., the number of people)

\[ \text{[Dalvi&S.'04]} \]

**Proof:** PP2CNF: \( F = (X_{i1} \lor Y_{j1}) \land (X_{i2} \lor Y_{j2}) \land \ldots \) reduce \( \#F \) to computing \( P(H_0) \)

By example:

\( F = (X_1 \lor Y_1) \land (X_1 \lor Y_2) \land (X_2 \lor Y_2) \)

**Probabilities** (tuples not shown have \( P=1 \))

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Smoker</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( x )</td>
<td>( 0.5 )</td>
<td></td>
</tr>
<tr>
<td>( x_1 )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( x_2 )</td>
<td>( 0.5 )</td>
<td></td>
</tr>
</tbody>
</table>

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Friend</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( x )</td>
<td>( y )</td>
<td>( P )</td>
</tr>
<tr>
<td>( x_1 )</td>
<td>( y_1 )</td>
<td>( 0 )</td>
</tr>
<tr>
<td>( x_1 )</td>
<td>( y_2 )</td>
<td>( 0 )</td>
</tr>
<tr>
<td>( x_2 )</td>
<td>( y_2 )</td>
<td>( 0 )</td>
</tr>
</tbody>
</table>

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Jogger</td>
<td></td>
</tr>
<tr>
<td>( y )</td>
<td>( P )</td>
</tr>
<tr>
<td>( y_1 )</td>
<td>( 0.5 )</td>
</tr>
<tr>
<td>( y_2 )</td>
<td>( 0.5 )</td>
</tr>
</tbody>
</table>
Our Problematic Clause

\[ H_0 = \forall x \forall y \text{Smoker}(x) \lor \text{Friend}(x,y) \lor \text{Jogger}(y) \]

**Theorem.** Computing \( P(H_0) \) is \#P-hard in the size of weight function \( w(.) \) (i.e., the number of people)

[Dalvi&S.’04]

**Proof:** PP2CNF: \( F = (X_{i1} \lor Y_{j1}) \land (X_{i2} \lor Y_{j2}) \land \ldots \) reduce \( \#F \) to computing \( P(H_0) \)

By example:

\[
F = (X_1 \lor Y_1) \land (X_1 \lor Y_2) \land (X_2 \lor Y_2)
\]

\( P(H_0) = P(F); \) hence \( P(H_0) \) is \#P-hard

<table>
<thead>
<tr>
<th>X</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_1 )</td>
<td>0.5</td>
</tr>
<tr>
<td>( x_2 )</td>
<td>0.5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_1 )</td>
<td>( y_1 )</td>
<td>0</td>
</tr>
<tr>
<td>( x_1 )</td>
<td>( y_2 )</td>
<td>0</td>
</tr>
<tr>
<td>( x_2 )</td>
<td>( y_2 )</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Y</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>( y_1 )</td>
<td>0.5</td>
</tr>
<tr>
<td>( y_2 )</td>
<td>0.5</td>
</tr>
</tbody>
</table>
What we know
What we know

1. Any d-D FO Circuit $Q$ admits efficient $P(Q)$ in the size of weight function $w(.)$
What we know

1. Any d-D FO Circuit $Q$ admits efficient $P(Q)$ in the size of weight function $w(.)$
2. Computing $P(H_0)$ is $\#P$-hard
What we know

1. Any d-D FO Circuit \( Q \) admits efficient \( P(Q) \) in the size of weight function \( w(.) \)

2. Computing \( P(H_0) \) is \( \#P \)-hard

3. Therefore \( H_0 \) has no d-D FO Circuit under standard complexity assumptions
What we know

1. Any d-D FO Circuit $Q$ admits efficient $P(Q)$ in the size of weight function $w(.)$
2. Computing $P(H_0)$ is $\#P$-hard
3. Therefore $H_0$ has no d-D FO Circuit under standard complexity assumptions

Next: This generalizes!
Background: Hierarchical Queries

\( \text{at}(x) = \text{set of atoms containing the variable } x \)

**Definition** \( Q \) is hierarchical if for all variables \( x, y \):

\[
\text{at}(x) \subseteq \text{at}(y) \quad \text{or} \quad \text{at}(x) \supseteq \text{at}(y) \quad \text{or} \quad \text{at}(x) \cap \text{at}(y) = \emptyset
\]
Background: Hierarchical Queries

$\text{at}(x) = \text{set of atoms containing the variable } x$

**Definition**  $Q$ is **hierarchical** if for all variables $x, y$:

$\text{at}(x) \subseteq \text{at}(y)$ or $\text{at}(x) \supseteq \text{at}(y)$ or $\text{at}(x) \cap \text{at}(y) = \emptyset$

Hierarchical

$Q = \forall x \: \forall y \: \forall z \: S(x,y) \lor T(x,z)$
Background: Hierarchical Queries

at(x) = set of atoms containing the variable x

**Definition** Q is **hierarchical** if for all variables x, y:

\[
\text{at}(x) \subseteq \text{at}(y) \quad \text{or} \quad \text{at}(x) \supseteq \text{at}(y) \quad \text{or} \quad \text{at}(x) \cap \text{at}(y) = \emptyset
\]

Hierarchical

Q = \( \forall x \ \forall y \ \forall z \ S(x,y) \lor T(x,z) \)

Non-hierarchical

\( H_0 = \forall x \ \forall y \ S(x) \lor F(x,y) \lor J(y) \)
The Small Dichotomy Theorem

**Theorem** Let $Q$ be *one clause, with no repeated symbols*
- If $Q$ is hierarchical, then $P(Q)$ is in $\text{PTIME}$.
- If $Q$ is not hierarchical then $P(Q)$ is $\#P$-hard.

[Dalvi&S.04]
The Small Dichotomy Theorem

**Theorem** Let \( Q \) be *one clause, with no repeated symbols*

- If \( Q \) is hierarchical, then \( P(Q) \) is in \( \text{PTIME} \).
- If \( Q \) is not hierarchical then \( P(Q) \) is \( \#P \)-hard.

[Dalvi&S.04]

**Corollary** Let \( Q \) be *one clause, with no repeated symbols*

- If \( Q \) is hierarchical, then \( Q \) has a d-D FO Circuit
- If \( Q \) is not hierarchical then \( Q \) has no d-D FO Circuit
  under standard complexity assumptions
The Small Dichotomy Theorem

**Theorem** Let $Q$ be *one clause, with no repeated symbols*
- If $Q$ is hierarchical, then $P(Q)$ is in $\text{PTIME}$.  
- If $Q$ is not hierarchical then $P(Q)$ is $\#P$-hard.

[Dalvi&S.04]

**Corollary** Let $Q$ be *one clause, with no repeated symbols*
- If $Q$ is hierarchical, then $Q$ has a $d$-$D$ FO Circuit
- If $Q$ is not hierarchical then $Q$ has no $d$-$D$ FO Circuit under standard complexity assumptions

Checking “$Q$ is hierarchical” is in $\text{AC}^0$ (expression complexity)
Compiling the $d$-$D$ FO Circuit is in $\text{PTIME}$
Proof

Hierarchical $\rightarrow$ PTIME
Proof

Hierarchical $\rightarrow$ PTIME

Case 1:

$Q = \begin{array}{c} \text{X} \\
\end{array}$

$\forall x$ must be decomposable
Proof

Hierarchical $\rightarrow$ PTIME

Case 1:

$Q = X$

$\forall x$ must be decomposable

Case 2:

$Q = Q_1 \lor Q_2$

$v$ must be decomposable
Proof

Hierarchical $\rightarrow$ PTIME

Case 1:

$Q = \cdots S(x, \ldots) \lor F(x, y, \ldots) \lor J(y, \ldots), \ldots$

$\forall x$ must be decomposable

Case 2:

$Q = Q_1 \lor Q_2$

$\lor$ must be decomposable

Non-hierarchical $\rightarrow$ \#P-hard

Reduction from $H_0$:
Overview

1. Propositional Refresher
2. Primer: A First-Order Tractable Language
3. Probabilistic Databases
4. Symmetric First-Order Model Counting
5. Lots of Pointers
Probabilistic Databases

• Tuple-independent probabilistic database

<table>
<thead>
<tr>
<th>Scientist</th>
<th>x</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>Erdos</td>
<td>0.9</td>
<td></td>
</tr>
<tr>
<td>Einstein</td>
<td>0.8</td>
<td></td>
</tr>
<tr>
<td>Pauli</td>
<td>0.6</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Coauthor</th>
<th>x</th>
<th>y</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>Erdos</td>
<td>Renyi</td>
<td>0.6</td>
<td></td>
</tr>
<tr>
<td>Einstein</td>
<td>Pauli</td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>Obama</td>
<td>Erdos</td>
<td>0.1</td>
<td></td>
</tr>
</tbody>
</table>

• Learned from the web, large text corpora, ontologies, etc., using **statistical** machine learning.

[Suciu’11]
Probabilistic Databases

∃x Coauthor(Einstein,x) ∧ Coauthor(Erdos,x)

• Conjunctive queries (CQ)

∃ + ∧
Probabilistic Databases

- Conjunctive queries (CQ)
  \( \exists x \ Coauthor(Einstein, x) \land Coauthor(Erdos, x) \)

- Unions of conjunctive queries (UCQ)
  \( \lor \text{of} \ \exists + \land \)
Probabilistic Databases

\[
\exists x \text{ Coauthor}(Einstein, x) \land \text{Coauthor}(Erdos, x)
\]

- Conjunctive queries (CQ)
  \[
  \exists + \land
  \]
- Unions of conjunctive queries (UCQ)
  \[
  \lor \text{of } \exists + \land
  \]
- Duality
  - Negation of CQ is monotone \(\forall\)-clause
  - Negation of UCQ is monotone \(\forall\)-CNF
Tuple-Independent Probabilistic DB

Probabilistic database D:

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>$p_1$</td>
</tr>
<tr>
<td>A</td>
<td>C</td>
<td>$p_2$</td>
</tr>
<tr>
<td>B</td>
<td>C</td>
<td>$p_3$</td>
</tr>
</tbody>
</table>
Tuple-Independent Probabilistic DB

Probabilistic database D:

Possible worlds semantics:

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>$p_1$</td>
</tr>
<tr>
<td>A</td>
<td>C</td>
<td>$p_2$</td>
</tr>
<tr>
<td>B</td>
<td>C</td>
<td>$p_3$</td>
</tr>
</tbody>
</table>

$p_1p_2p_3$
Tuple-Independent Probabilistic DB

Probabilistic database $D$:  

Possible worlds semantics:

<table>
<thead>
<tr>
<th>$x$</th>
<th>$y$</th>
<th>$P$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A$</td>
<td>$B$</td>
<td>$p_1$</td>
</tr>
<tr>
<td>$A$</td>
<td>$C$</td>
<td>$p_2$</td>
</tr>
<tr>
<td>$B$</td>
<td>$C$</td>
<td>$p_3$</td>
</tr>
</tbody>
</table>
Tuple-Independent Probabilistic DB

Probabilistic database D:

Possible worlds semantics:

\[(1-p_1)(1-p_2)(1-p_3)\]
Probabilistic Query Evaluation

\[ Q = \exists x \exists y \text{Scientist}(x) \land \text{Coauthor}(x, y) \]

\[ P(Q) = \]

<table>
<thead>
<tr>
<th>Scientist</th>
<th>x</th>
<th>y</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>p_1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>p_2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>p_3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Coauthor</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td>A</td>
</tr>
<tr>
<td>A</td>
</tr>
<tr>
<td>B</td>
</tr>
<tr>
<td>B</td>
</tr>
<tr>
<td>B</td>
</tr>
</tbody>
</table>
Probabilistic Query Evaluation

\[ Q = \exists x \exists y \text{ Scientist}(x) \land \text{Coauthor}(x, y) \]

\[ P(Q) = 1 - (1 - q_1)(1 - q_2) \]

<table>
<thead>
<tr>
<th>Scientist</th>
<th>x</th>
<th>y</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A</td>
<td>D</td>
<td>q_1</td>
</tr>
<tr>
<td>A</td>
<td>A</td>
<td>E</td>
<td>q_2</td>
</tr>
<tr>
<td>B</td>
<td>B</td>
<td>F</td>
<td>q_3</td>
</tr>
<tr>
<td>B</td>
<td>B</td>
<td>G</td>
<td>q_4</td>
</tr>
<tr>
<td>B</td>
<td>B</td>
<td>H</td>
<td>q_5</td>
</tr>
</tbody>
</table>
Probabilistic Query Evaluation

\[ Q = \exists x \exists y \text{Scientist}(x) \land \text{Coauthor}(x, y) \]

\[
P(Q) = p_1 \left[ 1 - (1 - q_1)(1 - q_2) \right]
\]

<table>
<thead>
<tr>
<th>Scientist</th>
<th>p</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>p_1</td>
</tr>
<tr>
<td>B</td>
<td>p_2</td>
</tr>
<tr>
<td>C</td>
<td>p_3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>D</td>
<td>q_1</td>
</tr>
<tr>
<td>A</td>
<td>E</td>
<td>q_2</td>
</tr>
<tr>
<td>B</td>
<td>F</td>
<td>q_3</td>
</tr>
<tr>
<td>B</td>
<td>G</td>
<td>q_4</td>
</tr>
<tr>
<td>B</td>
<td>H</td>
<td>q_5</td>
</tr>
</tbody>
</table>
Probabilistic Query Evaluation

\[ Q = \exists x \exists y \text{Scientist}(x) \land \text{Coauthor}(x,y) \]

\[ P(Q) = p_1 \left[ 1 - (1-q_1)(1-q_2) \right] \left[ 1 - (1-q_3)(1-q_4)(1-q_5) \right] \]

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>D</td>
<td>q_1</td>
</tr>
<tr>
<td>A</td>
<td>E</td>
<td>q_2</td>
</tr>
<tr>
<td>B</td>
<td>F</td>
<td>q_3</td>
</tr>
<tr>
<td>B</td>
<td>G</td>
<td>q_4</td>
</tr>
<tr>
<td>B</td>
<td>H</td>
<td>q_5</td>
</tr>
</tbody>
</table>
Probabilistic Query Evaluation

\[ Q = \exists x \exists y \text{Scientist}(x) \land \text{Coauthor}(x,y) \]

\[ P(Q) = p_1 \left[ 1 - (1 - q_1)(1 - q_2) \right] p_2 \left[ 1 - (1 - q_3)(1 - q_4)(1 - q_5) \right] \]

<table>
<thead>
<tr>
<th>Scientist</th>
<th>x</th>
<th>y</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A</td>
<td>D</td>
<td>( q_1 )</td>
</tr>
<tr>
<td>A</td>
<td>A</td>
<td>E</td>
<td>( q_2 )</td>
</tr>
<tr>
<td>B</td>
<td>B</td>
<td>F</td>
<td>( q_3 )</td>
</tr>
<tr>
<td>B</td>
<td>B</td>
<td>G</td>
<td>( q_4 )</td>
</tr>
<tr>
<td>B</td>
<td>B</td>
<td>H</td>
<td>( q_5 )</td>
</tr>
</tbody>
</table>
**Probabilistic Query Evaluation**

\[ Q = \exists x \exists y \text{Scientist}(x) \land \text{Coauthor}(x,y) \]

\[ P(Q) = 1 - \left\{ 1 - p_1 \left[ 1 -(1-q_1)(1-q_2) \right] \right\} \cdot \left\{ 1 - p_2 \left[ 1 -(1-q_3)(1-q_4)(1-q_5) \right] \right\} \]

<table>
<thead>
<tr>
<th>Scientist</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>p_1</td>
</tr>
<tr>
<td>B</td>
<td>p_2</td>
</tr>
<tr>
<td>C</td>
<td>p_3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Coauthor</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
</tr>
<tr>
<td>A</td>
</tr>
<tr>
<td>B</td>
</tr>
<tr>
<td>B</td>
</tr>
<tr>
<td>B</td>
</tr>
</tbody>
</table>
From Probabilities to WMC

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>p_1</td>
</tr>
<tr>
<td>A</td>
<td>C</td>
<td>p_2</td>
</tr>
<tr>
<td>B</td>
<td>C</td>
<td>p_3</td>
</tr>
</tbody>
</table>

\[(1-p_1)p_2p_3\]

\[(1-p_1)(1-p_2)(1-p_3)\]
From Probabilities to WMC

Friend

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>$p_1$</td>
</tr>
<tr>
<td>A</td>
<td>C</td>
<td>$p_2$</td>
</tr>
<tr>
<td>B</td>
<td>C</td>
<td>$p_3$</td>
</tr>
</tbody>
</table>

$P = (1-p_1)(1-p_2)(1-p_3)$
From Probabilities to WMC

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>y</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>B</td>
<td>p1</td>
</tr>
<tr>
<td>B</td>
<td>C</td>
<td>p2</td>
</tr>
<tr>
<td>A</td>
<td>C</td>
<td>p3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>w(Friend(x,y))</th>
<th>w(¬Friend(x,y))</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>w1 = p1</td>
<td>w1 = 1-p1</td>
</tr>
<tr>
<td>A</td>
<td>C</td>
<td>w2 = p2</td>
<td>w2 = 1-p2</td>
</tr>
<tr>
<td>B</td>
<td>C</td>
<td>w3 = p3</td>
<td>w3 = 1-p3</td>
</tr>
<tr>
<td>A</td>
<td>A</td>
<td>w4 = 0</td>
<td>w4 = 1</td>
</tr>
<tr>
<td>A</td>
<td>C</td>
<td>w5 = 0</td>
<td>w5 = 1</td>
</tr>
</tbody>
</table>

Also for missing tuples!
Lifted Inference Rules

Preprocess $Q$ (omitted)
Lifted Inference Rules

Preprocess $Q$ (omitted)

Evaluate Probability on FO Circuit*

<table>
<thead>
<tr>
<th>Expression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P(\neg Q) = 1 - P(Q)$</td>
<td>Negation</td>
</tr>
<tr>
<td>$P(Q_1 \land Q_2) = P(Q_1) \cdot P(Q_2)$</td>
<td>Decomposable $\land, \lor$</td>
</tr>
<tr>
<td>$P(Q_1 \lor Q_2) = 1 - (1 - P(Q_1))(1 - P(Q_2))$</td>
<td></td>
</tr>
<tr>
<td>$P(\forall z Q) = \prod_{A \in \text{Domain}} P(Q[A/z])$</td>
<td>Decomposable $\forall, \exists$</td>
</tr>
<tr>
<td>$P(\exists z Q) = 1 - \prod_{A \in \text{Domain}} (1 - P(Q[A/z]))$</td>
<td></td>
</tr>
<tr>
<td>$P(Q_1 \land Q_2) = P(Q_1) + P(Q_2) - 1$</td>
<td>Deterministic $\land, \lor$</td>
</tr>
<tr>
<td>$P(Q_1 \lor Q_2) = P(Q_1) + P(Q_2)$</td>
<td></td>
</tr>
<tr>
<td>$P(\forall z Q) = 1 - \sum_{A \in \text{Domain}} 1 - P(Q[A/z])$</td>
<td>Deterministic $\forall, \exists$</td>
</tr>
<tr>
<td>$P(\exists z Q) = \sum_{A \in \text{Domain}} P(Q[A/z])$</td>
<td></td>
</tr>
</tbody>
</table>

Decomposability

Determinism

[Suciu’11]
Lifted Inference Rules

Preprocess $Q$ (omitted)

\[
P(Q1 \land Q2) = P(Q1) + P(Q2) - P(Q1 \lor Q2)
\]
\[
P(Q1 \lor Q2) = P(Q1) + P(Q2) - P(Q1 \land Q2)
\]

Decomposability

Determinism

Inclusion/Exclusion

[Suciu’11]
Lifted Inference Rules

Preprocess $Q$ (omitted)

Evaluate Probability on FO Circuit*

- **Negation**
  - $P(-Q) = 1 - P(Q)$
  - $P(Q_1 \land Q_2) = P(Q_1)P(Q_2)$
  - $P(Q_1 \lor Q_2) = 1 - (1 - P(Q_1))(1 - P(Q_2))$

- **Decomposable $\land, \lor$**
  - $P(\forall z \ Q) = \prod_{A \in \text{Domain}} P(Q[A/z])$
  - $P(\exists z \ Q) = 1 - \prod_{A \in \text{Domain}} (1 - P(Q[A/z]))$
  - $P(Q_1 \land Q_2) = P(Q_1) + P(Q_2) - 1$
  - $P(Q_1 \lor Q_2) = P(Q_1) + P(Q_2)$

- **Deterministic $\land, \lor$**
  - $P(\forall z \ Q) = 1 - \sum_{A \in \text{Domain}} 1 - P(Q[A/z])$
  - $P(\exists z \ Q) = \sum_{A \in \text{Domain}} P(Q[A/z])$

**Decomposability**

**Determinism**

$P(Q_1 \land Q_2) = P(Q_1) + P(Q_2) - P(Q_1 \lor Q_2)$

$P(Q_1 \lor Q_2) = P(Q_1) + P(Q_2) - P(Q_1 \land Q_2)$

*Inclusion/Exclusion

Why?

[Suciu’11]
Background: \#P-hard Queries $H_k$

$H_0 = R(x) \lor S(x,y) \lor T(y)$

$H_1 = [R(x_0) \lor S(x_0,y_0)] \land [S(x_1,y_1) \lor T(y_1)]$

*Will drop $\forall$ to reduce clutter*
Background: $\#P$-hard Queries $H_k$

$$H_0 = R(x) \lor S(x,y) \lor T(y)$$

$$H_1 = [R(x_0) \lor S(x_0,y_0)] \land [S(x_1,y_1) \lor T(y_1)]$$

$$H_2 = [R(x_0) \lor S_1(x_0,y_0)] \land [S_1(x_1,y_1) \lor S_2(x_1,y_1)] \lor [S_2(x_2,y_2) \lor T(y_2)]$$

Will drop $\forall$ to reduce clutter
Background: \#P-hard Queries $H_k$

$H_0 = R(x) \lor S(x,y) \lor T(y)$

$H_1 = [R(x_0) \lor S(x_0,y_0)] \land [S(x_1,y_1) \lor T(y_1)]$

$H_2 = [R(x_0) \lor S_1(x_0,y_0)] \land [S_1(x_1,y_1) \lor S_2(x_1,y_1)] \lor [S_2(x_2,y_2) \lor T(y_2)]$

$H_3 = [R(x_0) \lor S_1(x_0,y_0)]$
$\land [S_1(x_1,y_1) \lor S_2(x_1,y_1)]$
$\land [S_2(x_2,y_2) \lor S_3(x_2,y_2)]$
$\land [S_3(x_3,y_3) \lor T(y_3)]$

$\ldots$

Will drop $\lor$ to reduce clutter
Background: \textbf{#P-hard Queries $H_k$}

\begin{align*}
H_0 &= R(x) \lor S(x,y) \lor T(y) \\
H_1 &= [R(x_0) \lor S(x_0,y_0)] \land [S(x_1,y_1) \lor T(y_1)] \\
H_2 &= [R(x_0) \lor S_1(x_0,y_0)] \land [S_1(x_1,y_1) \lor S_2(x_1,y_1)] \lor [S_2(x_2,y_2) \lor T(y_2)] \\
H_3 &= [R(x_0) \lor S_1(x_0,y_0)] \\
    & \quad \land [S_1(x_1,y_1) \lor S_2(x_1,y_1)] \\
    & \quad \land [S_2(x_2,y_2) \lor S_3(x_2,y_2)] \\
    & \quad \land [S_3(x_3,y_3) \lor T(y_3)] \\
\end{align*}

\textit{Theorem.} Every query $H_k$ is \textbf{#P-hard}

[Dalvi&S’12]
I/E and Cancellations

\[ Q_W = \left[ (R(x_0) \lor S_1(x_0,y_0)) \land (S_2(x_2,y_2) \lor S_3(x_2,y_2)) \right] \]
\[ \lor \left[ (R(x_0) \lor S_1(x_0,y_0)) \land (S_3(x_3,y_3) \lor T(y_3)) \right] \]
\[ \lor \left[ (S_1(x_1,y_1) \lor S_2(x_1,y_1)) \land (S_3(x_3,y_3) \lor T(y_3)) \right] \]
I/E and Cancellations

\[ Q_W = [(R(x_0) \lor S_1(x_0,y_0)) \land (S_2(x_2,y_2) \lor S_3(x_2,y_2))] \]
\[ \lor [(R(x_0) \lor S_1(x_0,y_0)) \land (S_3(x_3,y_3) \lor T(y_3))] \]
\[ \lor [(S_1(x_1,y_1) \lor S_2(x_1,y_1)) \land (S_3(x_3,y_3) \lor T(y_3))] \]

\[ P(Q_W) = P(Q_1) + P(Q_2) + P(Q_3) + \]
\[ - P(Q_1 \land Q_2) - P(Q_2 \land Q_3) - P(Q_1 \land Q_3) \]
\[ + P(Q_1 \land Q_2 \land Q_3) \]

[Suciu’11]
I/E and Cancellations

\[ Q_W = \left[ (R(x_0) \lor S_1(x_0,y_0)) \land (S_2(x_2,y_2) \lor S_3(x_2,y_2)) \right] \]
\[ \lor \left[ (R(x_0) \lor S_1(x_0,y_0)) \land (S_3(x_3,y_3) \lor T(y_3)) \right] \]
\[ \lor \left[ (S_1(x_1,y_1) \lor S_2(x_1,y_1)) \land (S_3(x_3,y_3) \lor T(y_3)) \right] \]

\[ P(Q_W) = P(Q_1) + P(Q_2) + P(Q_3) + \]
\[ - P(Q_1 \land Q_2) - P(Q_2 \land Q_3) - P(Q_1 \land Q_3) \]
\[ + P(Q_1 \land Q_2 \land Q_3) \]
\[ = H_3 (\#P\text{-hard} !) \]
I/E and Cancellations

\[
Q_W = [(R(x_0) \lor S_1(x_0, y_0)) \land (S_2(x_2, y_2) \lor S_3(x_2, y_2))] \\
\lor [(R(x_0) \lor S_1(x_0, y_0)) \land (S_3(x_3, y_3) \lor T(y_3))] \\
\lor [(S_1(x_1, y_1) \lor S_2(x_1, y_1)) \land (S_3(x_3, y_3) \lor T(y_3))]
\]

\[
P(Q_W) = P(Q_1) + P(Q_2) + P(Q_3) + \\
- P(Q_1 \land Q_2) - P(Q_2 \land Q_3) - P(Q_1 \land Q_3) \\
+ P(Q_1 \land Q_2 \land Q_3)
\]

Also = \(H_3\) (#\(P\)-hard !)

[Suciu’11]
I/E and Cancellations

\[ Q_w = [(R(x_0) \lor S_1(x_0, y_0)) \land (S_2(x_2, y_2) \lor S_3(x_2, y_2))] \]
\[ \lor [(R(x_0) \lor S_1(x_0, y_0)) \land (S_3(x_3, y_3) \lor T(y_3))] \]
\[ \lor [(S_1(x_1, y_1) \lor S_2(x_1, y_1)) \land (S_3(x_3, y_3) \lor T(y_3))] \]

\[ P(Q_w) = P(Q_1) + P(Q_2) + P(Q_3) + \]
\[ - P(Q_1 \land Q_2) - P(Q_2 \land Q_3) - P(Q_1 \land Q_3) + P(Q_1 \land Q_2 \land Q_3) \]
\[ = H_3 \quad (#P\text{-hard !}) \]

Need to cancel terms to compute the query in \textbf{PTIME}
Using Mobius’ function on the implication lattice of \( Q_w \)

[Suciu’11]
The Big Dichotomy Theorem

Call Q *liftable* if the rules don’t get stuck.

**Dichotomy Theorem** Fix a UCQ query Q.
1. If Q is *liftable*, then \( P(Q) \) is in \( \text{PTIME} \)
2. If Q is *not liftable*, then \( P(Q) \) is \( \#P \)-complete

[Dalvi’12]
The Big Dichotomy Theorem

Call Q *liftable* if the rules don’t get stuck.

**Dichotomy Theorem** Fix a UCQ query Q.
1. If Q is liftable, then $P(Q)$ is in $\text{PTIME}$
2. If Q is not liftable, then $P(Q)$ is $\#P$-complete

[Dalvi’12]

Lifted inference rules are complete for UCQ!
Open Problem

- For CQs w/o repeated symbols, $\text{PTIME } Q = \text{FO circuit language}$
- We need inclusion/exclusion to capture $\text{PTIME UCQs}$
- I/E is arithmetic operation $P(Q1) + P(Q2) - P(Q1 \lor Q2)$
Open Problem

- For CQs w/o repeated symbols, \( \text{PTIME } Q = \text{FO circuit language} \)
- We need inclusion/exclusion to capture \( \text{PTIME UCQs} \)
- I/E is arithmetic operation

\[ P(Q1) + P(Q2) - P(Q1 \lor Q2) \]

What is the logical equivalent of inclusion-exclusion?
What is the circuit language capturing \( \text{PTIME UCQs} \)?
Open Problem

• For CQs w/o repeated symbols, $\text{PTIME Q} = \text{FO circuit language}$
• We need inclusion/exclusion to capture $\text{PTIME UCQs}$
• $\text{I/E is arithmetic operation}$

What is the logical equivalent of inclusion-exclusion?
What is the circuit language capturing $\text{PTIME UCQs}$?

• It is not decision-DNNF! (see Beame)
Linear Data Complexity

\[ Q = \exists x \exists y \text{Scientist}(x) \land \text{Coauthor}(x, y) \]

\[ P(Q) = 1 - \prod_{A \in \text{Domain}} (1 - P(\text{Scientist}(A) \land \exists y \text{Coauthor}(A, y))) \]

\[ = 1 - (1 - P(\text{Scientist}(A) \land \exists y \text{Coauthor}(A, y))) \times (1 - P(\text{Scientist}(B) \land \exists y \text{Coauthor}(B, y))) \times (1 - P(\text{Scientist}(C) \land \exists y \text{Coauthor}(C, y))) \times (1 - P(\text{Scientist}(D) \land \exists y \text{Coauthor}(D, y))) \times (1 - P(\text{Scientist}(E) \land \exists y \text{Coauthor}(E, y))) \times (1 - P(\text{Scientist}(F) \land \exists y \text{Coauthor}(F, y))) \]

...
Linear Data Complexity

\[ Q = \exists x \exists y \text{Scientist}(x) \land \text{Coauthor}(x,y) \]

\[ P(Q) = 1 - \prod_{A \in \text{Domain}} (1 - P(\text{Scientist}(A) \land \exists y \text{Coauthor}(A,y))) \]

\[ = 1 - (1 - P(\text{Scientist}(A) \land \exists y \text{Coauthor}(A,y))) \]
\[ \times (1 - P(\text{Scientist}(B) \land \exists y \text{Coauthor}(B,y))) \]
\[ \times (1 - P(\text{Scientist}(C) \land \exists y \text{Coauthor}(C,y))) \]
\[ \times (1 - P(\text{Scientist}(D) \land \exists y \text{Coauthor}(D,y))) \]
\[ \times (1 - P(\text{Scientist}(E) \land \exists y \text{Coauthor}(E,y))) \]
\[ \times (1 - P(\text{Scientist}(F) \land \exists y \text{Coauthor}(F,y))) \]
\[ \vdots \]

No supporting facts in database!
Linear Data Complexity

\[ Q = \exists x \ \exists y \ \text{Scientist}(x) \land \text{Coauthor}(x,y) \]

\[ P(Q) = 1 - \prod_{A \in \text{Domain}} (1 - P(\text{Scientist}(A) \land \exists y \ \text{Coauthor}(A,y))) \]

\[ = 1 - (1 - P(\text{Scientist}(A) \land \exists y \ \text{Coauthor}(A,y))) \times (1 - P(\text{Scientist}(B) \land \exists y \ \text{Coauthor}(B,y))) \times (1 - P(\text{Scientist}(C) \land \exists y \ \text{Coauthor}(C,y))) \times (1 - P(\text{Scientist}(D) \land \exists y \ \text{Coauthor}(D,y))) \times (1 - P(\text{Scientist}(E) \land \exists y \ \text{Coauthor}(E,y))) \times (1 - P(\text{Scientist}(F) \land \exists y \ \text{Coauthor}(F,y))) \times \ldots \]

No supporting facts in database!

Probability 0

[Ceylan’16]
Linear Data Complexity

\[ Q = \exists x \exists y \text{Scientist}(x) \land \text{Coauthor}(x,y) \]

\[ P(Q) = 1 - \prod_{A \in \text{Domain}} (1 - P(\text{Scientist}(A) \land \exists y \text{Coauthor}(A,y))) \]

\[ = 1 - (1 - P(\text{Scientist}(A) \land \exists y \text{Coauthor}(A,y))) \]
\[ \times (1 - P(\text{Scientist}(B) \land \exists y \text{Coauthor}(B,y))) \]
\[ \times (1 - P(\text{Scientist}(C) \land \exists y \text{Coauthor}(C,y))) \]
\[ \times (1 - P(\text{Scientist}(D) \land \exists y \text{Coauthor}(D,y))) \]
\[ \times (1 - P(\text{Scientist}(E) \land \exists y \text{Coauthor}(E,y))) \]
\[ \times (1 - P(\text{Scientist}(F) \land \exists y \text{Coauthor}(F,y))) \]
\[ \ldots \]

\[ \rightarrow \text{No supporting facts in database!} \]
\[ \rightarrow \text{Probability 0} \]
\[ \rightarrow \text{Ignore these sub-queries!} \]

[Ceylan’16]
Linear Data Complexity

\[ Q = \exists x \exists y \text{Scientist}(x) \land \text{Coauthor}(x,y) \]

\[ P(Q) = 1 - \prod_{A \in \text{Domain}} (1 - P(\text{Scientist}(A) \land \exists y \text{Coauthor}(A,y))) \]

\[ = 1 - (1 - P(\text{Scientist}(A) \land \exists y \text{Coauthor}(A,y))) \times (1 - P(\text{Scientist}(B) \land \exists y \text{Coauthor}(B,y))) \times (1 - P(\text{Scientist}(C) \land \exists y \text{Coauthor}(C,y))) \times (1 - P(\text{Scientist}(D) \land \exists y \text{Coauthor}(D,y))) \times (1 - P(\text{Scientist}(E) \land \exists y \text{Coauthor}(E,y))) \times (1 - P(\text{Scientist}(F) \land \exists y \text{Coauthor}(F,y))) \times \ldots \]

No supporting facts in database!

Probability 0

Ignore these sub-queries!

Complexity linear time in database size!

[Ceylan’16]
Commercial Break

- Survey book (2011)

- IJCAI 2016 tutorial
  http://web.cs.ucla.edu/~guyvdb/talks/IJCAI16-tutorial/
Overview

1. Propositional Refresher

2. Primer: A First-Order Tractable Language

3. Probabilistic Databases

4. Symmetric First-Order Model Counting

5. Lots of Pointers
Simple Reasoning Problem

Probability that Card1 is Hearts? 1/4

[Van den Broeck; AAAI-KRR’15]
Model distribution by FOMC:

\[ \Delta = \forall p, \exists c, \text{Card}(p,c) \land \forall c, \exists p, \text{Card}(p,c) \land \forall p, \forall c, \forall c', \text{Card}(p,c) \land \text{Card}(p,c') \Rightarrow c = c' \]
Beyond NP Pipeline for #P

Reduce to propositional model counting:

[Van den Broeck 2015]
Beyond NP Pipeline for #P

Reduce to propositional model counting:

\[ \Delta = \text{Card}(A\heartsuit, p_1) \lor \ldots \lor \text{Card}(2\spadesuit, p_1) \]
\[ \text{Card}(A\heartsuit, p_2) \lor \ldots \lor \text{Card}(2\spadesuit, p_2) \]
\[ \ldots \]
\[ \text{Card}(A\heartsuit, p_1) \lor \ldots \lor \text{Card}(A\heartsuit, p_{52}) \]
\[ \text{Card}(K\heartsuit, p_1) \lor \ldots \lor \text{Card}(K\heartsuit, p_{52}) \]
\[ \ldots \]
\[ \lnot \text{Card}(A\heartsuit, p_1) \lor \lnot \text{Card}(A\heartsuit, p_2) \]
\[ \lnot \text{Card}(A\heartsuit, p_1) \lor \lnot \text{Card}(A\heartsuit, p_3) \]
\[ \ldots \]

[Van den Broeck 2015]
Beyond NP Pipeline for \#P

Reduce to propositional model counting:

\[ \Delta = \text{Card}(A♥, p_1) \lor \ldots \lor \text{Card}(2♣, p_1) \]
\[ \lor \text{Card}(A♥, p_2) \lor \ldots \lor \text{Card}(2♣, p_2) \]
\[ \ldots \]
\[ \lor \text{Card}(A♥, p_1) \lor \ldots \lor \text{Card}(A♥, p_{52}) \]
\[ \lor \text{Card}(K♥, p_1) \lor \ldots \lor \text{Card}(K♥, p_{52}) \]
\[ \ldots \]
\[ \lor \neg \text{Card}(A♥, p_1) \lor \neg \text{Card}(A♥, p_2) \]
\[ \lor \neg \text{Card}(A♥, p_1) \lor \neg \text{Card}(A♥, p_3) \]
\[ \ldots \]

What will happen?

[Van den Broeck 2015]
Deck of Cards Graphically
Deck of Cards Graphically

Card(♣, p_{52})
Deck of Cards Graphically

One model/\textit{perfect matching}
Deck of Cards Graphically

A♥

2♥

3♥

K♥
Deck of Cards Graphically

Card(K♥, p_{52})
Deck of Cards Graphically

Card(K♥, p_{52})

Model counting: How many *perfect* matchings?
What if I set

\[ w(\text{Card}(K\spadesuit, p_{52})) = 0? \]
What if I set
\[ w(\text{Card}(K\heartsuit, p_{52})) = 0? \]
Observations

• Weight function = bipartite graph
• # models = # perfect matchings
• Problem is \#P-complete! 😞
Observations

- Weight function = bipartite graph
- # models = # perfect matchings
- Problem is \#P-complete! 😞

No propositional WMC solver can handle cards problem efficiently!

[VdB’15]
Observations

- Weight function = bipartite graph
- # models = # perfect matchings
- Problem is \#P-complete! 😞

No propositional WMC solver can handle cards problem efficiently!

What is going on here?
Symmetric Weighted FOMC

No database! No literal-specific weights!

**Def.** A *weighted vocabulary* is $(\mathbf{R}, \mathbf{w})$, where

- $\mathbf{R} = (R_1, R_2, \ldots, R_k)$ = relational vocabulary
- $\mathbf{w} = (w_1, w_2, \ldots, w_k)$ = weights
- Implicit weights: $w(R_i(t)) = w_i$

Special case: $w_i = 1$ is model counting

Complexity in terms of domain size $n$
FOMC Inference Rules

- Simplification to $\exists, \forall$ rules:

  For example:
  \[
P(\forall z \ Q) = P(Q[C_1/z])|\text{Domain}|
  \]
FOMC Inference Rules

• Simplification to $\exists, \forall$ rules:

For example:
\[ P(\forall z \ Q) = P(Q[C_1/z])^{\text{Domain}} \]

• A powerful new inference rule: *atom counting*
  Only possible with symmetric weights
  Intuition: **Remove unary relations**

[VdB’11]
Deterministic Decomposable FO NNF

\[ \Delta = \forall x, y \in \text{People}: \text{Smokes}(x) \land \text{Friends}(x, y) \Rightarrow \text{Smokes}(y) \]
$\Delta = \forall x, y \in \text{People}: \text{Smokes}(x) \land \text{Friends}(x, y) \Rightarrow \text{Smokes}(y)$

[Van den Broeck 2013]
Deterministic Decomposable FO NNF

\[ \Delta = \forall x, y \in \text{People}: \, \text{Smokes}(x) \land \text{Friends}(x, y) \Rightarrow \text{Smokes}(y) \]
\[ \Delta = \forall x, y \in \text{People}: \text{Smokes}(x) \land \text{Friends}(x,y) \Rightarrow \text{Smokes}(y) \]
Deterministic Decomposable FO NNF

Δ = ∀x, y ∈ People: Smokes(x) ∧ Friends(x, y) ⇒ Smokes(y)

[Van den Broeck 2013]
Deterministic Decomposable FO NNF

\[ \Delta = \forall x, y \in \text{People}: \ Smokes(x) \land \text{Friends}(x, y) \Rightarrow \text{Smokes}(y) \]
First-Order Model Counting: Example

$\Delta = \forall x, y \in \text{People}: \text{Smokes}(x) \land \text{Friends}(x, y) \Rightarrow \text{Smokes}(y)$
First-Order Model Counting: Example

\[ \Delta = \forall x, y \in \textbf{People}: \text{Smokes}(x) \land \text{Friends}(x, y) \implies \text{Smokes}(y) \]

- If we know \( D \) precisely: who smokes, and there are \( k \) smokers?

**Database:**

<table>
<thead>
<tr>
<th>Name</th>
<th>Smokes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>1</td>
</tr>
<tr>
<td>Bob</td>
<td>0</td>
</tr>
<tr>
<td>Charlie</td>
<td>0</td>
</tr>
<tr>
<td>Dave</td>
<td>1</td>
</tr>
<tr>
<td>Eve</td>
<td>0</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

\[ k \]

\[ n-k \]
First-Order Model Counting: Example

\[ \Delta = \forall x, y \in \text{People}: \text{Smokes}(x) \land \text{Friends}(x, y) \Rightarrow \text{Smokes}(y) \]

- If we know \( D \) precisely: who smokes, and there are \( k \) smokers?

<table>
<thead>
<tr>
<th>Database</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smokes(Alice) = 1</td>
</tr>
<tr>
<td>Smokes(Bob) = 0</td>
</tr>
<tr>
<td>Smokes(Charlie) = 0</td>
</tr>
<tr>
<td>Smokes(Dave) = 1</td>
</tr>
<tr>
<td>Smokes(Eve) = 0</td>
</tr>
<tr>
<td>...</td>
</tr>
</tbody>
</table>

[Van den Broeck 2015]
First-Order Model Counting: Example

\[ \Delta = \forall x, y \in \text{People}: \text{Smokes}(x) \land \text{Friends}(x, y) \Rightarrow \text{Smokes}(y) \]

- If we know \( \mathbf{D} \) precisely: who smokes, and there are \( k \) smokers?

**Database:**
- Smokes(Alice) = 1
- Smokes(Bob) = 0
- Smokes(Charlie) = 0
- Smokes(Dave) = 1
- Smokes(Eve) = 0
- ...

[Van den Broeck 2015]
First-Order Model Counting: Example

\[ \Delta = \forall x, y \in \text{People}: \text{Smokes}(x) \land \text{Friends}(x, y) \Rightarrow \text{Smokes}(y) \]

- If we know \( D \) precisely: who smokes, and there are \( k \) smokers?

**Database:**
- Smokes(Alice) = 1
- Smokes(Bob) = 0
- Smokes(Charlie) = 0
- Smokes(Dave) = 1
- Smokes(Eve) = 0
- ...

[Van den Broeck 2015]
First-Order Model Counting: Example

\[ \Delta = \forall x, y \in \text{People}: \ Smokes(x) \land Friends(x,y) \Rightarrow Smokes(y) \]

- If we know D precisely: who smokes, and there are \( k \) smokers?

**Database:**
- Smokes(Alice) = 1
- Smokes(Bob) = 0
- Smokes(Charlie) = 0
- Smokes(Dave) = 1
- Smokes(Eve) = 0
- ...

[Van den Broeck 2015]
First-Order Model Counting: Example

\[ \Delta = \forall x, y \in \textbf{People}: \text{Smokes}(x) \land \text{Friends}(x, y) \Rightarrow \text{Smokes}(y) \]

- If we know \( D \) precisely: who smokes, and there are \( k \) smokers?

**Database:**
- Smokes(Alice) = 1
- Smokes(Bob) = 0
- Smokes(Charlie) = 0
- Smokes(Dave) = 1
- Smokes(Eve) = 0
- ...
First-Order Model Counting: Example

\[ \Delta = \forall x, y \in \text{People}: \text{Smokes}(x) \land \text{Friends}(x, y) \Rightarrow \text{Smokes}(y) \]

- If we know \( D \) precisely: who smokes, and there are \( k \) smokers?

**Database:**
- Smokes(Alice) = 1
- Smokes(Bob) = 0
- Smokes(Charlie) = 0
- Smokes(Dave) = 1
- Smokes(Eve) = 0
- ...

[Van den Broeck 2015]
First-Order Model Counting: Example

\[ \Delta = \forall x, y \in \text{People}: \text{Smokes}(x) \land \text{Friends}(x, y) \Rightarrow \text{Smokes}(y) \]

- If we know D precisely: who smokes, and there are \( k \) smokers?

**Database:**
- Smokes(Alice) = 1
- Smokes(Bob) = 0
- Smokes(Charlie) = 0
- Smokes(Dave) = 1
- Smokes(Eve) = 0
- ...

[Van den Broeck 2015]
First-Order Model Counting: Example

\[ \Delta = \forall x, y \in \textbf{People}: \text{Smokes}(x) \land \text{Friends}(x,y) \Rightarrow \text{Smokes}(y) \]

- If we know \( D \) precisely: who smokes, and there are \( k \) smokers?

\textbf{Database:}
- Smokes(Alice) = 1
- Smokes(Bob) = 0
- Smokes(Charlie) = 0
- Smokes(Dave) = 1
- Smokes(Eve) = 0
- ...

[Van den Broeck 2015]
First-Order Model Counting: Example

\[ \Delta = \forall x, y \in \textbf{People}: \text{Smokes}(x) \land \text{Friends}(x, y) \Rightarrow \text{Smokes}(y) \]

- If we know \( D \) precisely: who smokes, and there are \( k \) smokers?

**Database:**
- \( \text{Smokes}(\text{Alice}) = 1 \)
- \( \text{Smokes}(\text{Bob}) = 0 \)
- \( \text{Smokes}(\text{Charlie}) = 0 \)
- \( \text{Smokes}(\text{Dave}) = 1 \)
- \( \text{Smokes}(\text{Eve}) = 0 \)
- ...

\[ \Rightarrow 2^{n^2 - k(n - k)} \] models

[Van den Broeck 2015]
First-Order Model Counting: Example

\[ \Delta = \forall x, y \in \text{People}: \text{Smokes}(x) \land \text{Friends}(x, y) \Rightarrow \text{Smokes}(y) \]

- If we know \( D \) precisely: who smokes, and there are \( k \) smokers?

  Database:
  - \( \text{Smokes}(\text{Alice}) = 1 \)
  - \( \text{Smokes}(\text{Bob}) = 0 \)
  - \( \text{Smokes}(\text{Charlie}) = 0 \)
  - \( \text{Smokes}(\text{Dave}) = 1 \)
  - \( \text{Smokes}(\text{Eve}) = 0 \)
  - ...

  \[ 2^{n^2 - k(n-k)} \] models

- If we know that there are \( k \) smokers?
First-Order Model Counting: Example

\[ \Delta = \forall x, y \in \textbf{People}: \text{Smokes}(x) \land \text{Friends}(x, y) \Rightarrow \text{Smokes}(y) \]

- If we know \( \mathbf{D} \) precisely: who smokes, and there are \( k \) smokers?

  Database:
  
  - Smokes(Alice) = 1
  - Smokes(Bob) = 0
  - Smokes(Charlie) = 0
  - Smokes(Dave) = 1
  - Smokes(Eve) = 0
  ...

  \[ \Rightarrow 2^{n^2 - k(n-k)} \] models

- If we know that there are \( k \) smokers?

  \[ \Rightarrow \binom{n}{k} 2^{n^2 - k(n-k)} \] models

[Van den Broeck 2015]
First-Order Model Counting: Example

\[ \Delta = \forall x, y \in \textbf{People}: \text{Smokes}(x) \land \text{Friends}(x,y) \Rightarrow \text{Smokes}(y) \]

- If we know \( D \) precisely: who smokes, and there are \( k \) smokers?

  \[ \text{Database:} \]
  
  Smokes(Alice) = 1  
  Smokes(Bob) = 0  
  Smokes(Charlie) = 0  
  Smokes(Dave) = 1  
  Smokes(Eve) = 0  
  ...  

  \[ \Rightarrow 2^{n^2 - k(n-k)} \] models

- If we know that there are \( k \) smokers?  
  \[ \Rightarrow \binom{n}{k} 2^{n^2 - k(n-k)} \] models

- In total...

[Van den Broeck 2015]
First-Order Model Counting: Example

\[ \Delta = \forall x, y \in \text{People}: \text{Smokes}(x) \land \text{Friends}(x, y) \Rightarrow \text{Smokes}(y) \]

- If we know \( D \) precisely: who smokes, and there are \( k \) smokers?

**Database:**
- \( \text{Smokes}(\text{Alice}) = 1 \)
- \( \text{Smokes}(\text{Bob}) = 0 \)
- \( \text{Smokes}(\text{Charlie}) = 0 \)
- \( \text{Smokes}(\text{Dave}) = 1 \)
- \( \text{Smokes}(\text{Eve}) = 0 \)
- ...

\[ \rightarrow 2^{n^2 - k(n-k)} \] models

- If we know that there are \( k \) smokers?

\[ \rightarrow \binom{n}{k} 2^{n^2 - k(n-k)} \] models

- In total...

\[ \rightarrow \sum_{k=0}^{n} \binom{n}{k} 2^{n^2 - k(n-k)} \] models

[Van den Broeck 2015]
Main Positive Result: $\text{FO}^2$

- $\text{FO}^2 = \text{FO restricted to two variables}$
- “The graph has a path of length 10”:
  $$\exists x \exists y (R(x,y) \land \exists x (R(y,x) \land \exists y (R(x,y) \land \ldots)))$$

- Theorem: Compilation algorithm to \text{FO} d-DNNF is complete for $\text{FO}^2$
- Model counting for $\text{FO}^2$ in PTIME domain complexity
Main Negative Results

Domain complexity:

• There is an FO formula $Q$ s.t. $\text{FOMC}(Q, n)$ is $\#P_1$-hard

• There is a $Q$ in $\text{FO}^3$ s.t. $\text{FOMC}(Q, n)$ is $\#P_1$-hard

• There exists a conjunctive query $Q$ s.t. symmetric $\text{WFOMC}(Q, n)$ is $\#P_1$-hard

• There exists a positive clause $Q$ w.o. ‘$\neq$’ s.t. symmetric $\text{WFOMC}(Q, n)$ is $\#P_1$-hard

Therefore, no FO d-DNNF exists (unless…)

😊
Tractable Classes

[VD; NIPS’11], [VD et al.; KR’14], [Gribkoff, VdB, Suciu; UAI’15], [Beame, VdB, Gribkoff, Suciu; PODS’15], etc.
Tractable Classes

Δ = ∀x,y,z, Friends(x,y) ∧ Friends(y,z) ⇒ Friends(x,z)

[VdB; NIPS’11], [VdB et al.; KR’14], [Gribkoff, VdB, Suciu; UAI’15], [Beame, VdB, Gribkoff, Suciu; PODS’15], etc.
Skolemization for WFOMC

\[ \Delta = \forall p, \exists c, \text{Card}(p, c) \]
Skolemization for WFOMC

\[ \Delta = \forall p, \exists c, \text{Card}(p,c) \]

\[ \Delta' = \forall p, \forall c, \text{Card}(p,c) \Rightarrow S(p) \]
Skolemization for WFOMC

\[ \Delta = \forall p, \exists c, \text{Card}(p,c) \]

\[ \Delta' = \forall p, \forall c, \text{Card}(p,c) \Rightarrow S(p) \]

\[ w(S) = 1 \quad \text{and} \quad w(\neg S) = -1 \]

Skolem predicate

[VdB'14]
Skolemization for WFOMC

\[ \Delta = \forall p, \exists c, \text{Card}(p,c) \]

\[ \Delta' = \forall p, \forall c, \text{Card}(p,c) \Rightarrow S(p) \]

Consider one position \( p \):

\[ \exists c, \text{Card}(p,c) = \text{true} \]

\[ \exists c, \text{Card}(p,c) = \text{false} \]

\[ w(S) = 1 \quad \text{and} \quad w(\neg S) = -1 \]

Skolemization

Skolem predicate

[VdB’14]
Skolemization for WFOMC

\[ \Delta = \forall p, \exists c, \text{Card}(p,c) \]

\[ \Delta' = \forall p, \forall c, \text{Card}(p,c) \Rightarrow S(p) \]

Consider one position \( p \):

\[ \exists c, \text{Card}(p,c) = \text{true} \]

\[ S(p) = \text{true} \]

\[ \exists c, \text{Card}(p,c) = \text{false} \]

\[ w(S) = 1 \text{ and } w(\neg S) = -1 \]

[Skolem predicate]

Also model of \( \Delta \), weight * 1

[VdB'14]
Skolemization for WFOMC

\[ \Delta = \forall p, \exists c, \text{Card}(p,c) \]

Skolemization

\[ \Delta' = \forall p, \forall c, \text{Card}(p,c) \Rightarrow S(p) \]

Consider one position \( p \):

\[ \exists c, \text{Card}(p,c) = \text{true} \]

\[ S(p) = \text{true} \]

Also model of \( \Delta \), weight \( * 1 \)

\[ w(S) = 1 \text{ and } w(\neg S) = -1 \]

\[ \exists c, \text{Card}(p,c) = \text{false} \]

\[ S(p) = \text{true} \]

No model of \( \Delta \), weight \( * 1 \)

\[ S(p) = \text{false} \]

No model of \( \Delta \), weight \( * -1 \)

Skolem predicate

Extra models

Cancel out

[VdB’14]
Resolution for WFOMC

\[ \Delta = \forall x \forall y \ (R(x) \lor \neg S(x,y)) \land \forall x \forall y \ (S(x,y) \lor T(y)) \]

Rules stuck…

Resolution on \( S(x,y) \):

\[ \forall x \forall y \ (R(x) \lor T(y)) \]

Add resolvent:

\[ \Delta = \forall x \forall y \ (R(x) \lor \neg S(x,y)) \land \forall x \forall y \ (S(x,y) \lor T(y)) \land \forall x \forall y \ (R(x) \lor T(y)) \]

Now apply I/E!
Compilation Rules

• Standard rules
  – Shannon decomposition (DPLL)
  – Detect decomposability
  – Etc.

• FO Shannon decomposition:

[Van den Broeck 2013]
∀p, ∃c, Card(p,c)
∀c, ∃p, Card(p,c)
∀p, ∀c, ∀c’, Card(p,c) ∧ Card(p,c’) ⇒ c = c’
Playing Cards Revisited

\[ \forall p, \exists c, \text{Card}(p, c) \]
\[ \forall c, \exists p, \text{Card}(p, c) \]
\[ \forall p, \forall c, \forall c', \text{Card}(p, c) \land \text{Card}(p, c') \Rightarrow c = c' \]

\[ \#\text{SAT} = \sum_{k=0}^{n} \binom{n}{k} \sum_{l=0}^{n} \binom{n}{l} (l + 1)^k (-1)^{2n-k-l} = n! \]

Playing Cards Revisited

∀p, ∃c, Card(p,c)
∀c, ∃p, Card(p,c)
∀p, ∀c, ∀c', Card(p,c) ∧ Card(p,c') ⇒ c = c'

#SAT = \sum_{k=0}^{n} \binom{n}{k} \sum_{l=0}^{n} \binom{n}{l} (l + 1)^k (-1)^{2n-k-l} = n!

Computed in time polynomial in n

Overview

1. Propositional Refresher
2. Primer: A First-Order Tractable Language
3. Probabilistic Databases
4. Symmetric First-Order Model Counting
5. Lots of Pointers
Pointers

- Work on first-order knowledge compilation in `90s
  - Henry Kautz

- Factored Databases
  - Dan Olteanu

- New inference rules for symmetric counting (domain recursion)
  - Guy
More Pointers

• PTIME UCQ queries and circuit lower bounds

  Paul Beame

• Compiling first-order database queries to propositional circuits

  Dan Olteanu  Dan Suciu  Pierre Bourhis  Pierre Senellart
More Pointers

• Database fixed-parameter tractability

• Colour refinement to detect first-order structure

• Probabilistic database preference models and triangle queries
Statistical Relational Learning

Markov Logic

\[ \text{Smokes}(x) \land \text{Friends}(x,y) \Rightarrow \text{Smokes}(y) \]
Statistical Relational Learning

Markov Logic

3.14  \( \text{Smokes}(x) \land \text{Friends}(x, y) \Rightarrow \text{Smokes}(y) \)

FOL Sentence

\[ \forall x, y, F(x, y) \Leftrightarrow [ \text{Smokes}(x) \land \text{Friends}(x, y) \Rightarrow \text{Smokes}(y) ] \]

Weight Function

\[ w(\text{Smokes}) = 1 \]
\[ w(\neg \text{Smokes}) = 1 \]
\[ w(\text{Friends}) = 1 \]
\[ w(\neg \text{Friends}) = 1 \]
\[ w(F) = 3.14 \]
\[ w(\neg F) = 1 \]

[Van den Broeck, PhD’13]
Statistical Relational Learning

Markov Logic

3.14  \( \text{Smokes}(x) \land \text{Friends}(x,y) \Rightarrow \text{Smokes}(y) \)

Weight Function

\[
\begin{align*}
    w(\text{Smokes}) &= 1 \\
    w(\neg \text{Smokes}) &= 1 \\
    w(\text{Friends}) &= 1 \\
    w(\neg \text{Friends}) &= 1 \\
    w(\text{F}) &= 3.14 \\
    w(\neg \text{F}) &= 1
\end{align*}
\]

FOL Sentence

\( \forall x,y, F(x,y) \iff [ \text{Smokes}(x) \land \text{Friends}(x,y) \Rightarrow \text{Smokes}(y) ] \)

Compile?

First-Order d-DNNF Circuit

[Van den Broeck, PhD’13]
Statistical Relational Learning

Markov Logic

\[ 3.14 \quad \text{Smokes}(x) \land \text{Friends}(x,y) \Rightarrow \text{Smokes}(y) \]

Weight Function

- \( w(\text{Smokes}) = 1 \)
- \( w(\neg \text{Smokes}) = 1 \)
- \( w(\text{Friends}) = 1 \)
- \( w(\neg \text{Friends}) = 1 \)
- \( w(F) = 3.14 \)
- \( w(\neg F) = 1 \)

FOL Sentence

\[ \forall x,y, F(x,y) \iff [ \text{Smokes}(x) \land \text{Friends}(x,y) \Rightarrow \text{Smokes}(y) ] \]

Compile?

First-Order d-DNNF Circuit

[Van den Broeck, PhD’13]
Statistical Relational Learning

Markov Logic

3.14 Smokes(x) ∧ Friends(x,y) ⇒ Smokes(y)

Weight Function

\[
\begin{align*}
    w(\text{Smokes}) &= 1 \\
    w(\neg \text{Smokes}) &= 1 \\
    w(\text{Friends}) &= 1 \\
    w(\neg \text{Friends}) &= 1 \\
    w(F) &= 3.14 \\
    w(\neg F) &= 1 
\end{align*}
\]

FOL Sentence

\[\forall x,y, F(x,y) ⇔ [ \text{Smokes}(x) ∧ \text{Friends}(x,y) ⇒ \text{Smokes}(y) ]\]

Compile?

First-Order d-DNNF Circuit

Domain

Alice
Bob
Charlie

Z = WFOMC = 1479.85

[Van den Broeck, PhD’13]
Statistical Relational Learning

Markov Logic

FOL Sentence

Weight Function

First-Order d-DNNF Circuit

Compile?

Domain

Evaluation in time polynomial in domain size!

3.14 \[\text{Smokes}(x) \land \text{Friends}(x,y) \Rightarrow \text{Smokes}(y)\]

\[\forall x,y, F(x,y) \leftrightarrow [ \text{Smokes}(x) \land \text{Friends}(x,y) \Rightarrow \text{Smokes}(y) ]\]

\[w(\text{Smokes}) = 1\]
\[w(\neg \text{Smokes}) = 1\]
\[w(\text{Friends}) = 1\]
\[w(\neg \text{Friends}) = 1\]
\[w(F) = 3.14\]
\[w(\neg F) = 1\]

Alice
Bob
Charlie

\[Z = \text{WFOMC} = 1479.85\]

[Van den Broeck, PhD’13]
Statistical Relational Learning

Markov Logic

Weight Function

\[ w(\text{Smokes}) = 1 \]
\[ w(\neg \text{Smokes}) = 1 \]
\[ w(\text{Friends}) = 1 \]
\[ w(\neg \text{Friends}) = 1 \]
\[ w(\text{F}) = 3.14 \]
\[ w(\neg \text{F}) = 1 \]

FOL Sentence

\[ \forall x, y, F(x,y) \iff [ \text{Smokes}(x) \land \text{Friends}(x,y) \Rightarrow \text{Smokes}(y) ] \]

 Compile?

First-Order d-DNNF Circuit

Domain

Alice
Bob
Charlie

Evaluation in time polynomial in domain size!

[Van den Broeck, PhD’13]
FO² is liftable!

Properties
- Smokes(x)
- Gender(x)
- Young(x)
- Tall(x)

Properties
- Smokes(y)
- Gender(y)
- Young(y)
- Tall(y)
$\mathbf{FO^2}$ is liftable!

**Properties**
- Smokes($x$)
- Gender($x$)
- Young($x$)
- Tall($x$)

**Relations**
- Friends($x,y$)
- Colleagues($x,y$)
- Family($x,y$)
- Classmates($x,y$)

**Properties**
- Smokes($y$)
- Gender($y$)
- Young($y$)
- Tall($y$)
“Smokers are more likely to be friends with other smokers.”
“Colleagues of the same age are more likely to be friends.”
“People are either family or friends, but never both.”
“If X is family of Y, then Y is also family of X.”
“If X is a parent of Y, then Y cannot be a parent of X.”
More Pointers

• Lifted machine learning

• Open-world probabilistic databases
Generalized Model Counting

Probability Distribution

\[ = \]

Logic

+ 

Weights
Generalized Model Counting

Probability Distribution

\[ \text{Logic} \]
\[ + \]
\[ \text{Weights} \]

Logical Syntax

Model-theoretic Semantics

\[ + \]

Weight function \( w(.) \)
Weighted Model Integration

\[
\text{Probability Distribution} = \text{SMT(LRA)} + \text{Weights}
\]

[Belle et al. IJCAI’15, UAI’15]
Weighted Model Integration

Probability Distribution

\[ = \]

\[ \text{SMT(LRA)} \]

\[ + \]

Weights

0 \leq \text{height} \leq 200
0 \leq \text{weight} \leq 200
0 \leq \text{age} \leq 100
\text{age} < 1 \Rightarrow \text{height} + \text{weight} \leq 90

\[ + \]

w(\text{height}) = \text{height} - 10
w(\neg \text{height}) = 3 \times \text{height}^2
w(\neg \text{weight}) = 5

\[ \ldots \]

[Belle et al. IJCAI'15, UAI'15]
Weighted Model Integration

Probability Distribution

\[ \text{SMT(LRA)} + \text{Weights} \]

Scott Sanner

- \(0 \leq \text{height} \leq 200\)
- \(0 \leq \text{weight} \leq 200\)
- \(0 \leq \text{age} \leq 100\)
- \(\text{age} < 1 \Rightarrow \text{weight} \leq 90\)

- \(w(\text{height}) = \text{height} - 10\)
- \(w(\neg \text{height}) = 3^2 \text{height}^2\)
- \(w(\neg \text{weight}) = 5\)

...
Probabilistic Programming

Probability Distribution = Logic Programs + Weights

[Fierens et al., TPLP’15]
Probabilistic Programming

Probability Distribution

= Logic Programs

+ Weights

path(X,Y) :-
    edge(X,Y).
path(X,Y) :-
    edge(X,Z), path(Z,Y).

[Fierens et al., TPLP’15]
Probabilistic Programming

Probability Distribution

Logic Programs

Weights

path(X,Y) :-
  edge(X,Y).
path(X,Y) :-
  edge(X,Z), path(Z,Y).

Fierens et al., TPLP'15

Wannes Meert

[Fierens et al., TPLP’15]
Conclusions

• Determinism and decomposability generalize to first-order logic
• First-order model counting unifies
  – Probabilistic databases
  – High-level statistical AI models
• Fascinating computational complexity questions
• Requires dedicated first-order solvers
QUESTIONS?

THE FIRST ORDER NEEDS YOU