Outline

• Part 1: Motivation
• Part 2: Probabilistic Databases
• Part 3: Weighted Model Counting
• Part 4: Lifted Inference for WFOMC
• Part 5: Completeness of Lifted Inference
• Part 6: Query Compilation
• Part 7: Symmetric Lifted Inference Complexity
• Part 8: Open-World Probabilistic Databases
• Part 9: Discussion & Conclusions
Defining Lifted Inference

- Informal:
  Exploit symmetries, Reason at first-order level, Reason about groups of objects, Scalable inference, High-level probabilistic reasoning, etc.

- A formal definition: **Domain-lifted inference**

  Inference runs in time polynomial in the number of objects in the domain.

  - Polynomial in \#people, \#webpages, \#cards
  - **Not** polynomial in \#predicates, \#formulas, \#logical variables
  - Related to data complexity in databases

[VdB’11, Jaeger’12]
Defining Lifted Inference

- **Informal:**
  Exploit symmetries, Reason at first-order level, Reason about groups of objects, Scalable inference, High-level probabilistic reasoning, etc. [Poole’03, etc.]

- **A formal definition:** **Domain-lifted inference**

[VdB’11, Jaeger’12]
Defining Lifted Inference

- Informal:
  Exploit symmetries, Reason at first-order level, Reason about groups of objects, Scalable inference, High-level probabilistic reasoning, etc. [Poole’03, etc.]

- A formal definition: **Domain-lifted inference**

- Alternative in this tutorial:
  Lifted inference = \exists Query Plan = \exists FO Compilation

[VdB’11, Jaeger’12]
Asymmetric WFOMC Rules

Preprocess $Q$ (omitted from this talk; see [Suciu’11]), then apply these rules (some have preconditions)

$$\text{WMC}(\neg \Delta) = Z - \text{WMC}(\Delta)$$

Negation

Normalization constant $Z$ (easy to compute)
Asymmetric WFOMC Rules

Preprocess $Q$ (omitted from this talk; see [Suciu’11]), then apply these rules (some have preconditions)

- $\text{WMC}(\neg \Delta) = Z - \text{WMC}(\Delta)$
- $\text{WMC}(\Delta_1 \land \Delta_2) = \text{WMC}(\Delta_1) \times \text{WMC}(\Delta_2)$
- $\text{WMC}(\Delta_1 \lor \Delta_2) = Z - (Z_1 - \text{WMC}(\Delta_1)) \times (Z_2 - \text{WMC}(\Delta_2))$

Negation

Normalization constant $Z$
(easy to compute)

Independent join / union
Asymmetric WFOMC Rules

Preprocess $Q$ (omitted from this talk; see [Suciu’11]), then apply these rules (some have preconditions)

$$\text{WMC}(\neg \Delta) = Z - \text{WMC}(\Delta)$$

Negation

Normalization constant $Z$ (easy to compute)

$$\text{WMC}(\Delta_1 \land \Delta_2) = \text{WMC}(\Delta_1) \times \text{WMC}(\Delta_2)$$

$$\text{WMC}(\Delta_1 \lor \Delta_2) = Z - (Z_1 - \text{WMC}(\Delta_1))*(Z_2 - \text{WMC}(\Delta_2))$$

Independent join / union

$$\text{WMC}(\exists z \, \Delta) = Z - \Pi_{C \in \text{Domain}} (Z_C - \text{WMC}(\Delta[C/z]))$$

$$\text{WMC}(\forall z \, \Delta) = \Pi_{C \in \text{Domain}} \text{WMC}(\Delta[C/z])$$

Independent project
Asymmetric WFOMC Rules

Preprocess $Q$ (omitted from this talk; see [Suciu’11]), then apply these rules (some have preconditions)

- **Negation**
  \[ WMC(\neg \Delta) = Z - WMC(\Delta) \]

- **Independent join/union**
  \[ WMC(\Delta_1 \land \Delta_2) = WMC(\Delta_1) \times WMC(\Delta_2) \]
  \[ WMC(\Delta_1 \lor \Delta_2) = Z - (Z_1 - WMC(\Delta_1)) \times (Z_2 - WMC(\Delta_2)) \]

- **Independent project**
  \[ WMC(\exists z \Delta) = Z - \Pi_{C \in \text{Domain}} (Z_C - WMC(\Delta[C/z])) \]
  \[ WMC(\forall z \Delta) = \Pi_{C \in \text{Domain}} WMC(\Delta[C/z]) \]

- **Inclusion/exclusion**
  \[ WMC(\Delta_1 \land \Delta_2) = WMC(\Delta_1) + WMC(\Delta_2) - WMC(\Delta_1 \lor \Delta_2) \]
  \[ WMC(\Delta_1 \lor \Delta_2) = WMC(\Delta_1) + WMC(\Delta_2) - WMC(\Delta_1 \land \Delta_2) \]
Symmetric WFOMC Rules

• Simplification to *independent project*:

\[ \Delta[C_1/x], \Delta[C_2/x], \ldots \text{ are independent} \]

\[
\begin{align*}
WMC(\exists z \Delta) &= Z - (Z_{C_1} - WMC(\Delta[C_1/z]))|\text{Domain}| \\
WMC(\forall z \Delta) &= WMC(\Delta[C_1/z])|\text{Domain}| 
\end{align*}
\]

[VdB’11]
Symmetric WFOMC Rules

• Simplification to *independent project*:

\[
\begin{align*}
\text{If } \Delta[C_1/x], \Delta[C_2/x], \ldots \text{ are independent} & \quad \Rightarrow \\
\text{WMC}(\exists z \Delta) &= Z - (Z_{C_1} - \text{WMC}(\Delta[C_1/z]))|_{\text{Domain}} \\
\text{WMC}(\forall z \Delta) &= \text{WMC}(\Delta[C_1/z])|_{\text{Domain}} 
\end{align*}
\]

• A powerful new inference rule: *atom counting*

Only possible with symmetric weights.

Intuition: **Remove unary relations**

The workhorse of Symmetric WFOMC

[VdB’11]
WFOMC Inference: Example

- FO-Model Counting: $w(R) = w(\neg R) = 1$
- Apply inference rules backwards (step 4-3-2-1)
WFOMC Inference: Example

- FO-Model Counting: \( w(R) = w(\neg R) = 1 \)
- Apply inference rules backwards (step 4-3-2-1)

4. \( \Delta = (\text{Stress(Alice)} \Rightarrow \text{Smokes(Alice)}) \)

Domain = \{Alice\}
WFOMC Inference: Example

- FO-Model Counting: $w(R) = w(\neg R) = 1$
- Apply inference rules backwards (step 4-3-2-1)

4. $\Delta = (\text{Stress(Alice)} \Rightarrow \text{Smokes(Alice)})$  
   Domain = $\{\text{Alice}\}$  
   $\rightarrow 3$ models
WFOMC Inference: Example

- FO-Model Counting: \( w(R) = w(\neg R) = 1 \)
- Apply inference rules backwards (step 4-3-2-1)

4. \[ \Delta = (\text{Stress}(\text{Alice}) \Rightarrow \text{Smokes}(\text{Alice})) \]

\[
\text{WMC}(\neg\text{Stress}(\text{Alice}) \lor \text{Smokes}(\text{Alice}))) = \\
= Z - \text{WMC}(\text{Stress}(\text{Alice})) \times \text{WMC}(\neg\text{Smokes}(\text{Alice})) \\
= 4 - 1 \times 1 = 3 \text{ models}
\]

Domain = {Alice}
WFOMC Inference: Example

• FO-Model Counting: $w(R) = w(\neg R) = 1$
• Apply inference rules backwards (step 4-3-2-1)

4. \[ \Delta = (\text{Stress}(\text{Alice}) \Rightarrow \text{Smokes}(\text{Alice})) \]
   Domain = \{\text{Alice}\}
   
   \[
   \text{WMC}(\neg\text{Stress}(\text{Alice}) \lor \text{Smokes}(\text{Alice}))) = \\
   = Z - \text{WMC}(\text{Stress}(\text{Alice})) \times \text{WMC}(\neg\text{Smokes}(\text{Alice})) \\
   = 4 - 1 \times 1 = 3 \text{ models}
   
3. \[ \Delta = \forall x, (\text{Stress}(x) \Rightarrow \text{Smokes}(x)) \]
   Domain = \{\text{n people}\}
WFOMC Inference: Example

- FO-Model Counting: \( w(R) = w(\neg R) = 1 \)
- Apply inference rules backwards (step 4-3-2-1)

\[ \Delta = (\text{Stress(Alice) } \Rightarrow \text{Smokes(Alice)}) \]

Domain = \{Alice\}

\[ \text{WMC}(\neg \text{Stress(Alice)} \lor \text{Smokes(Alice)}) = Z - \text{WMC}(\text{Stress(Alice)}) \times \text{WMC}(\neg \text{Smokes(Alice)}) = 4 - 1 \times 1 = 3 \text{ models} \]

\[ \Delta = \forall x, (\text{Stress(x) } \Rightarrow \text{Smokes(x)}) \]

Domain = \{n people\}

\[ \rightarrow 3^n \text{ models} \]
WFOMC Inference: Example

3. \[ \Delta = \forall x, (\text{Stress}(x) \Rightarrow \text{Smokes}(x)) \]

\[ \rightarrow 3^n \text{ models} \]

Domain = \{n people\}
WFOMC Inference: Example

3. \[ \Delta = \forall x, (\text{Stress}(x) \Rightarrow \text{Smokes}(x)) \]

\[ \rightarrow 3^n \text{ models} \]

Domain = \{n people\}

2. \[ \Delta = \forall y, (\text{ParentOf}(y) \land \text{Female} \Rightarrow \text{MotherOf}(y)) \]

D = \{n people\}
WFOMC Inference: Example

3. \[ \Delta = \forall x, (\text{Stress}(x) \Rightarrow \text{Smokes}(x)) \]
   \[ \text{Domain} = \{n \text{ people}\} \]
   \[ \rightarrow 3^n \text{ models} \]

2. \[ \Delta = \forall y, (\text{ParentOf}(y) \land \text{Female} \Rightarrow \text{MotherOf}(y)) \]
   \[ D = \{n \text{ people}\} \]
   If Female = true?
   \[ \Delta = \forall y, (\text{ParentOf}(y) \Rightarrow \text{MotherOf}(y)) \]
   \[ \rightarrow 3^n \text{ models} \]
### WFOMC Inference: Example

<table>
<thead>
<tr>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.</td>
<td>$\Delta = \forall y, (\text{ParentOf}(y) \land \text{Female} \Rightarrow \text{MotherOf}(y))$</td>
</tr>
<tr>
<td></td>
<td>$D = {n \text{ people}}$</td>
</tr>
<tr>
<td></td>
<td>If Female = true? $\Delta = \forall y, (\text{ParentOf}(y) \Rightarrow \text{MotherOf}(y))$ $\rightarrow 3^n$ models</td>
</tr>
<tr>
<td></td>
<td>If Female = false? $\Delta = \text{true}$ $\rightarrow 4^n$ models</td>
</tr>
<tr>
<td>3.</td>
<td>$\Delta = \forall x, (\text{Stress}(x) \Rightarrow \text{Smokes}(x))$</td>
</tr>
<tr>
<td></td>
<td>$\rightarrow 3^n$ models</td>
</tr>
<tr>
<td></td>
<td>Domain = ${n \text{ people}}$</td>
</tr>
</tbody>
</table>
WFOMC Inference: Example

3. $\Delta = \forall x, (\text{Stress}(x) \Rightarrow \text{Smokes}(x))$
   Domain = \{n people\}
   $\rightarrow 3^n \text{ models}$

2. $\Delta = \forall y, (\text{ParentOf}(y) \wedge \text{Female} \Rightarrow \text{MotherOf}(y))$
   D = \{n people\}
   If Female = true?
   $\Delta = \forall y, (\text{ParentOf}(y) \Rightarrow \text{MotherOf}(y))$
   $\rightarrow 3^n \text{ models}$
   If Female = false?
   $\Delta = \text{true}$
   $\rightarrow 4^n \text{ models}$
   $\rightarrow 3^n + 4^n \text{ models}$
WFOMC Inference: Example

3. \[ \Delta = \forall x, (\text{Stress}(x) \Rightarrow \text{Smokes}(x)) \]
   \[ \text{Domain} = \{n \text{ people}\} \]
   \[ \rightarrow 3^n \text{ models} \]

2. \[ \Delta = \forall y, (\text{ParentOf}(y) \land \text{Female} \Rightarrow \text{MotherOf}(y)) \]
   \[ D = \{n \text{ people}\} \]
   \[ \text{WMC}(\Delta) = \text{WMC}(\neg \text{Female} \lor \forall y, (\text{ParentOf}(y) \Rightarrow \text{MotherOf}(y))) \]
   \[ = 2 \times 2^n \times 2^n - (2 - 1) \times (2^n \times 2^n - \text{WMC}(\forall y, (\text{ParentOf}(y) \Rightarrow \text{MotherOf}(y)))) \]
   \[ = 2 \times 4^n - (4^n - 3^n) \]
   \[ \rightarrow 3^n + 4^n \text{ models} \]
WFOMC Inference: Example

1. \[ \Delta = \forall x, y, (\text{ParentOf}(x,y) \land \text{Female}(x) \Rightarrow \text{MotherOf}(x,y)) \]  

   \[ \text{Domain} = \{n \text{ people}\} \]

   \[ \rightarrow 3^n \text{ models} \]

2. \[ \Delta = \forall y, (\text{ParentOf}(y) \land \text{Female} \Rightarrow \text{MotherOf}(y)) \]  

   \[ \text{D} = \{n \text{ people}\} \]

   \[ \text{WMC}(\Delta) = \text{WMC}(\neg \text{Female} \lor \forall y, (\text{ParentOf}(y) \Rightarrow \text{MotherOf}(y))) \]

   \[ = 2 \cdot 2^n \cdot 2^n - (2 - 1) \cdot (2^n \cdot 2^n - \text{WMC}(\forall y, (\text{ParentOf}(y) \Rightarrow \text{MotherOf}(y)))) \]

   \[ = 2 \cdot 4^n - (4^n - 3^n) \]

   \[ \rightarrow 3^n + 4^n \text{ models} \]

3. \[ \Delta = \forall x, (\text{Stress}(x) \Rightarrow \text{Smokes}(x)) \]  

   \[ \text{Domain} = \{n \text{ people}\} \]

   \[ \rightarrow 3^n \text{ models} \]
WFOMC Inference: Example

3. \[ \Delta = \forall x, \ (\text{Stress}(x) \Rightarrow \text{Smokes}(x)) \]
   \[ \rightarrow 3^n \text{ models} \]

2. \[ \Delta = \forall y, \ (\text{ParentOf}(y) \land \text{Female} \Rightarrow \text{MotherOf}(y)) \]
   \[ \text{Domain} = \{n \text{ people}\} \]
   \[ \text{WMC}(\Delta) = \text{WMC}(\neg \text{Female} \lor \forall y, \ (\text{ParentOf}(y) \Rightarrow \text{MotherOf}(y))) \]
   \[ = 2 \times 2^n \times 2^n - (2 - 1) \times (2^n \times 2^n - \text{WMC}(\forall y, \ (\text{ParentOf}(y) \Rightarrow \text{MotherOf}(y)))) \]
   \[ = 2 \times 4^n - (4^n - 3^n) \]
   \[ \rightarrow 3^n + 4^n \text{ models} \]

1. \[ \Delta = \forall x,y, \ (\text{ParentOf}(x,y) \land \text{Female}(x) \Rightarrow \text{MotherOf}(x,y)) \]
   \[ \text{D} = \{n \text{ people}\} \]
   \[ \rightarrow (3^n + 4^n)^n \text{ models} \]
Atom Counting: Example

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

Domain = \{n people\}
Atom Counting: Example

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

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

**Database:**

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

<table>
<thead>
<tr>
<th>Smokes</th>
<th>Friends</th>
<th>Smokes</th>
</tr>
</thead>
<tbody>
<tr>
<td>k</td>
<td></td>
<td>k</td>
</tr>
<tr>
<td>n-k</td>
<td></td>
<td>n-k</td>
</tr>
</tbody>
</table>
Atom Counting: Example

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

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

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

Domain = \{n people\}
Atom Counting: Example

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

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

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

Domain = \( \{n \text{ people}\} \)
Atom Counting: Example

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

Domain = \{n \text{ people}\}

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

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

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

- If we know 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
- ...

\[ \text{Domain} = \{ n \text{ people} \} \]
Atom Counting: Example

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

Domain = \{n people\}

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

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

![Diagram of smoke relationships]
Atom Counting: Example

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

Domain = \{\text{n people}\}

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

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

$\Delta = \forall x, y, (\text{Smokes}(x) \land \text{Friends}(x,y) \Rightarrow \text{Smokes}(y))$

- If we know precisely who smokes, and there are $k$ smokers?

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

Domain = {n people}
Atom Counting: Example

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

Domain = \{n people\}

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

**Database:**

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

Diagram: Flowchart showing relationships between 'Smokes', 'Friends', and 'Smokes'.
Atom Counting: Example

If we know precisely who smokes, and there are $k$ smokers?

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

$\Delta = \forall x, y, (\text{Smokes}(x) \land \text{Friends}(x, y) \Rightarrow \text{Smokes}(y))$

Domain = \{n people\}

$$\Rightarrow 2^{n^2 - k(n-k)}$$ models
Atom Counting: Example

$$\Delta = \forall x,y, (\text{Smokes}(x) \land \text{Friends}(x,y) \Rightarrow \text{Smokes}(y))$$

Domain = \{n \text{ people}\}

- If we know precisely who smokes, and there are $k$ smokers?

Database:

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

$$\Rightarrow 2^{n^2 - k(n-k)} \text{ models}$$

- If we know that there are $k$ smokers?
Atom Counting: Example

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

- If we know 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?

\[ \binom{n}{k} 2^{n^2 - k(n-k)} \] models
Atom Counting: Example

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

Domain = \{ n \text{ people} \}

- If we know precisely who smokes, and there are \( k \) smokers?
  
  **Database:**
  
  - Smokes(Alice) = 1
  - Smokes(Bob) = 0
  - Smokes(Charlie) = 0
  - Smokes(Dave) = 1
  - Smokes(Eve) = 0
  ...

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

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

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

- In total...
Atom Counting: Example

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

Domain = \{n people\}

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

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

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

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

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

- In total...

\[ \sum_{k=0}^{n} \binom{n}{k} 2^{n^2-k(n-k)} \] models
Augment Rules with Logical Rewritings

[Suciu’11]
Augment Rules with Logical Rewritings

1. Remove constants (shattering)

\[ \Delta = \forall x \ (\text{Friend}(\text{Alice}, x) \lor \text{Friend}(x, \text{Bob})) \]
Augment Rules with Logical Rewritings

1. Remove constants (shattering)

\[ \Delta = \forall x \ (\text{Friend}(\text{Alice}, x) \lor \text{Friend}(x, \text{Bob})) \]

\[ \Delta = \forall x \ (F_1(x) \lor F_2(x)) \land (F_3 \lor F_4) \land (F_4 \lor F_5) \]

F_1(x) = \text{Friend}(\text{Alice}, x)
F_2(x) = \text{Friend}(x, \text{Bob})
F_3 = \text{Friend}(\text{Alice}, \text{Alice})
F_4 = \text{Friend}(\text{Alice}, \text{Bob})
F_5 = \text{Friend}(\text{Bob}, \text{Bob})
Augment Rules with Logical Rewritings

1. Remove constants (shattering)

\[ \Delta = \forall x \ (\text{Friend}(\text{Alice}, x) \lor \text{Friend}(x, \text{Bob})) \]

\[ \Rightarrow \Delta = \forall x \ (F_1(x) \lor F_2(x)) \land (F_3 \lor F_4) \land (F_4 \lor F_5) \]

\[ F_1(x) = \text{Friend}(\text{Alice}, x) \]
\[ F_2(x) = \text{Friend}(x, \text{Bob}) \]
\[ F_3 = \text{Friend}(\text{Alice}, \text{Alice}) \]
\[ F_4 = \text{Friend}(\text{Alice}, \text{Bob}) \]
\[ F_5 = \text{Friend}(\text{Bob}, \text{Bob}) \]

2. “Rank” variables (= occur in the same order in each atom)

\[ \Delta = (\text{Friend}(x, y) \lor \text{Enemy}(x, y)) \land (\text{Friend}(x, y) \lor \text{Enemy}(y, x)) \]

Wrong order

[Suciu’11]
Augment Rules with Logical Rewritings

1. Remove constants (shattering)

\[ \Delta = \forall x \ (\text{Friend}(\text{Alice}, x) \lor \text{Friend}(x, \text{Bob})) \]

\[ \Delta = \forall x \ (F_1(x) \lor F_2(x)) \land (F_3 \lor F_4) \land (F_4 \lor F_5) \]

2. “Rank” variables (= occur in the same order in each atom)

\[ \Delta = (\text{Friend}(x,y) \lor \text{Enemy}(x,y)) \land (\text{Friend}(x,y) \lor \text{Enemy}(y,x)) \]

Wrong order

\[ \Delta = (F_1(x,y) \lor E_1(x,y)) \land (F_1(x,y) \lor E_3(x,y)) \land (F_2(x) \lor E_2(x)) \land (F_3(x,y) \lor E_3(x,y)) \land (F_3(x,y) \lor E_1(x,y)) \]

\[ F_1(x) = \text{Friend}(\text{Alice},x) \]
\[ F_2(x) = \text{Friend}(x,\text{Bob}) \]
\[ F_3 = \text{Friend}(\text{Alice}, \text{Alice}) \]
\[ F_4 = \text{Friend}(\text{Alice},\text{Bob}) \]
\[ F_5 = \text{Friend}(\text{Bob},\text{Bob}) \]

[Sucić’11]
Augment Rules with Logical Rewritings

3. Perform Resolution [Gribkoff’14]

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

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!
Augment Rules with Logical Rewritings

4. Skolemization [VdB’14]

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

Inference rules assume one type of quantifier!

Mix \( \forall/\exists \) in encodings of MLNs with quantifiers and probabilistic programs

Datalog

\[
\text{smokes}(X) :- \text{friends}(X,Y), \text{smokes}(Y).
\]

FOL

\[ \Delta = \forall x, \text{Smokes}(x) \Leftrightarrow \exists y, \text{Friends}(x,y), \text{Smokes}(y). \]

Skolemization

Input: Mix \( \forall/\exists \)  
Output: Only \( \forall \)

BUT: cannot introduce Skolem constants or functions!

\[ \forall p, \text{Card}(p, \text{S}(p)) \]
Skolemization: Example

$\Delta = \forall p, \exists c, \text{Card}(p,c)$
Skolemization: Example

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

$\Delta' = \forall p, \forall c, \text{Card}(p,c) \Rightarrow S(p)$
Skolemization: Example

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

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

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

Skolem predicate

[VdB’14]
Skolemization: Example

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

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

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

[VdB'14]
Skolemization: Example

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

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

Consider one position \( p \):

- \( \exists c, \text{Card}(p,c) = \text{true} \) \implies \( S(p) = \text{true} \)
- \( \exists c, \text{Card}(p,c) = \text{false} \)

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

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

[VdB'14]
Skolemization: Example

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

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

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

Extra models Cancel out

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

| Markov Logic | 3.14 Smokes(x) ∧ Friends(x,y) ⇒ Smokes(y) |
First-Order Knowledge Compilation

Markov Logic

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

Weight Function

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

FOL Sentence

∀x,y, F(x,y) ⇔ [ Smokes(x) ∧ Friends(x,y) ⇒ Smokes(y) ]
First-Order Knowledge Compilation

Markov Logic

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

Weight Function

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

FOL Sentence

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

Compile?

First-Order d-DNNF Circuit

[Vdb’11,’13]
First-Order Knowledge Compilation

Markov Logic

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

Weight Function

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

FOL Sentence

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

Compile?

First-Order d-DNNF Circuit

\( Z = \text{WFOMC} = 1479.85 \)

[Vdb’11,’13]
First-Order Knowledge Compilation

Markov Logic

FOL Sentence

∀x,y, F(x,y) ⇔ [ Smokes(x) ∧ Friends(x,y) ⇒ Smokes(y) ]

Compile?

First-Order d-DNNF Circuit

Evaluation in time polynomial in domain size

Z = WFOMC = 1479.85

Weight Function

w(Smokes)=1
w(¬Smokes )=1
w(Friends )=1
w(¬Friends )=1
w(F)=exp(3.14)
w(¬F)=1

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

Domain

Alice
Bob
Charlie

[wdb’11,’13]
First-Order Knowledge Compilation

Markov Logic

Weight Function

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

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

Domain

Alice
Bob
Charlie

Evaluation in time polynomial in domain size

Domain-lifted!

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

[Vdb’11,’13]
Negation Normal Form

[Darwiche'01]
Decomposable NNF

[Darwiche’01]
Deterministic Decomposable NNF

[Darwiche’01]
Deterministic Decomposable NNF

Weighted Model Counting

[Darwiche'01]
Deterministic Decomposable NNF

Weighted Model Counting and much more!

[Darwiche'01]
First-Order NNF

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

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

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

\[ \forall X, X \in \text{People} : \text{belgian}(X) \Rightarrow \text{likes}(X, \text{chocolate}) \]

[VdB’13]
First-Order NNF = Query Plan

\[ \forall X, X \in \text{People} : \text{belgian}(X) \Rightarrow \text{likes}(X, \text{chocolate}) \]
Deterministic Decomposable FO NNF

\[ \forall X, X \in \text{People} : \text{belgian}(X) \Rightarrow \text{likes}(X, chocolate) \]

Weighted Model Counting

[VdB’13]
Deterministic Decomposable FO NNF

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

Weighted Model Counting

\( \Pr(\text{belgian}) \times \Pr(\text{likes}) + \Pr(\neg \text{belgian}) \)

[VdB’13]
Deterministic Decomposable FO NNF

∀X, X ∈ People : belgian(X) ⇒ likes(X, chocolate)

Weighted Model Counting

|People| = (Pr(belgian) × Pr(likes)) + Pr(¬belgian)

[VdB’13]
Symmetric WFOMC on FO NNF

\[
U(\alpha) = \begin{cases} 
0 & \text{when } \alpha = \text{false} \\
1 & \text{when } \alpha = \text{true} \\
0.5 & \text{when } \alpha \text{ is a literal} \\
U(\ell_1) \times \cdots \times U(\ell_n) & \text{when } \alpha = \ell_1 \land \cdots \land \ell_n \\
U(\ell_1) + \cdots + U(\ell_n) & \text{when } \alpha = \ell_1 \lor \cdots \lor \ell_n \\
\prod_{i=1}^{n} U(\beta\{X/x_i\}) & \text{when } \alpha = \forall X \in \tau, \beta \text{ and } x_1, \ldots, x_n \text{ are the objects in } \tau. \\
\sum_{i=1}^{n} U(\beta\{X/x_i\}) & \text{when } \alpha = \exists X \in \tau, \beta \text{ and } x_1, \ldots, x_n \text{ are the objects in } \tau. \\
\prod_{i=0}^{\left|\tau\right|} U(\beta\{X/x_i\})^{\left|\tau\right|} & \text{when } \alpha = \forall X \subseteq \tau, \beta, \text{ and } x_i \text{ is any subset of } \tau \text{ such that } |x_i| = i. \\
\sum_{i=0}^{\left|\tau\right|} \left(\binom{\left|\tau\right|}{i}\right) \cdot U(\beta\{X/x_i\}) & \text{when } \alpha = \exists X \subseteq \tau, \beta, \text{ and } x_i \text{ is any subset of } \tau \text{ such that } |x_i| = i.
\end{cases}
\]

Complexity polynomial in domain size!
Polynomial in NNF size for bounded depth.

[VdB’13]
How to do first-order knowledge compilation?
Deterministic Decomposable FO NNF

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

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

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

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

[VdB’13]
Deterministic Decomposable FO NNF

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

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

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

• FO Shannon decomposition:

\[
\exists D \\
D \subseteq \text{People}
\]

\[
\forall X \\
X \in D
\]

\[
\forall X \\
X \in \text{People} \land X \notin D
\]

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

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

[VdB’13]
Let us automate this:

- **Relational** model

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

- **Lifted** probabilistic inference algorithm
Why not do propositional WMC?

Reduce to propositional model counting:
Why not do propositional WMC?

Reduce to propositional model counting:

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

[VdB’15]
Why not do propositional WMC?

Reduce to propositional model counting:

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

What will happen?

[VdB’15]
Deck of Cards Graphically

A♥

2♥

3♥

K♥
Deck of Cards Graphically

Card(K♥, p_{52})

[VdB’15]
Deck of Cards Graphically

One model/\textit{perfect matching}

[VdB'15]
Deck of Cards Graphically

[VdB’15]
Deck of Cards Graphically

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

Model counting: How many *perfect matchings*?

Card(K♥,p_{52})

[VdB’15]
Deck of Cards Graphically

What if I set $w(\text{Card}(K\heartsuit, p_{52})) = 0$?
Deck of Cards Graphically

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

[VdB’15]
Deck of Cards Graphically

What if I set can set any asymmetric weight function?

[VdB’15]
Observations

• Asymmetric weight function can remove edge
  Encode any bigraph

• Counting models = perfect matchings
• Problem is \#P-complete! 😞

• All non-lifted WMC solvers efficiently handle asymmetric weights
• No solver does cards problem efficiently!

Later: Power of lifted vs. ground inference and complexities
Playing Cards Revisited

Let us automate this:

- **Relational** model

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

- **Lifted** probabilistic inference algorithm
Playing Cards Revisited

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

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

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

∀p, ∀c, Card(p,c) ⇒ S_1(p)
∀c, ∀p, Card(p,c) ⇒ S_2(c)
∀p, ∀c, ∀c', Card(p,c) ∧ Card(p,c') ⇒ c = c'

Skolemization
Playing Cards Revisited

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

\[ \downarrow \quad \text{Skolemization} \]

\[ \forall p, \forall c, \text{Card}(p,c) \Rightarrow S_1(p) \]
\[ \forall c, \forall p, \text{Card}(p,c) \Rightarrow S_2(c) \]
\[ \forall p, \forall c, \forall c', \text{Card}(p,c) \land \text{Card}(p,c') \Rightarrow c = c' \]

\[ w(S_1) = 1 \text{ and } w(\neg S_1) = -1 \]
\[ w(S_2) = 1 \text{ and } w(\neg S_2) = -1 \]
∀p, ∃c, Card(p,c)
∀c, ∃p, Card(p,c)
∀p, ∀c, ∀c’, Card(p,c) ∧ Card(p,c’) ⇒ c = c’

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

\[ w(S_1) = 1 \text{ and } w(\neg S_1) = -1 \]
\[ w(S_2) = 1 \text{ and } w(\neg S_2) = -1 \]
Playing Cards Revisited

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

\[ \Rightarrow \]

\[ \bullet \bullet \bullet \text{Skolemization} \]

\[ \forall p, \forall c, \text{Card}(p,c) \Rightarrow S_1(p) \]
\[ \forall c, \forall p, \text{Card}(p,c) \Rightarrow S_2(c) \]
\[ \forall p, \forall c, \forall c', \text{Card}(p,c) \land \text{Card}(p,c') \Rightarrow c = c' \]

\[ \Rightarrow \]

\[ \bullet \bullet \bullet \text{Atom counting} \]

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

\[ w(S_1) = 1 \text{ and } w(\neg S_1) = -1 \]
\[ w(S_2) = 1 \text{ and } w(\neg S_2) = -1 \]
∀p, Ǝc, Card(p,c)
∀c, Ǝp, Card(p,c)
∀p, ∀c, ∀c’, Card(p,c) & Card(p,c’) ⇒ c = c’

∀p, ∀c, Card(p,c) ⇒ S1(p)
∀c, ∀p, Card(p,c) ⇒ S2(c)
∀p, ∀c, ∀c’, Card(p,c) & Card(p,c’) ⇒ c = c’

w(S1) = 1 and w(¬S1) = -1
w(S2) = 1 and w(¬S2) = -1

∀p, ∀c, ∀c’, Card(p,c) & Card(p,c’) ⇒ c = c’

∀-Rule

[VdB’15]
∀p, ∃c, Card(p,c)
∀c, ∃p, Card(p,c)
∀p, ∀c, ∀c', Card(p,c) ∧ Card(p,c') ⇒ c = c'

- - - Skolemization

∀p, ∀c, Card(p,c) ⇒ S_1(p)
∀c, ∀p, Card(p,c) ⇒ S_2(c)
∀p, ∀c, ∀c', Card(p,c) ∧ Card(p,c') ⇒ c = c'

- - - Atom counting

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

- - - ∀-Rule

∀c, ∀c', Card(c) ∧ Card(c') ⇒ c = c'

w(S_1) = 1 and w(¬S_1) = -1
w(S_2) = 1 and w(¬S_2) = -1

Playing Cards Revisited

[VD815]
∀p, ∃c, Card(p,c)  
∀c, ∃p, Card(p,c)  
∀p, ∀c, ∀c', Card(p,c) ∧ Card(p,c') ⇒ c = c'

Skolemization

∀p, ∀c, Card(p,c) ⇒ S_1(p)  
∀c, ∀p, Card(p,c) ⇒ S_2(c)  
∀p, ∀c, ∀c', Card(p,c) ∧ Card(p,c') ⇒ c = c'

Atom counting

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

∀-Rule

∀c, ∀c', Card(c) ∧ Card(c') ⇒ c = c'

w(S_1) = 1 and w(¬S_1) = -1  
w(S_2) = 1 and w(¬S_2) = -1

Playing Cards Revisited

[VdB’15]
Playing Cards Revisited

Let us automate this:

- **Lifted** probabilistic inference algorithm

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

Computed in time polynomial in \( n \)
Summary Lifted Inference

• By definition: PTIME data complexity
  Also: \( \exists \) FO compilation = \( \exists \) Query Plan
• However: only works for “liftable” queries
• Preprocessing based on logical rewriting
• The rules: Deceptively simple: the only surprising rules are I/E and atom counting
• Rules are captured by a query plan or first-order NNF circuit