First-Order Knowledge Compilation for Probabilistic Reasoning

Guy Van den Broeck

based on joint work with Adnan Darwiche, Dan Suciu, and many others
MOTIVATION 1
A Simple Reasoning Problem

Probability that Card1 is Hearts?

[Van den Broeck; AAAI-KRR’15]
A Simple Reasoning Problem

Probability that Card1 is Hearts? 1/4
A Simple Reasoning Problem

Probability that Card52 is Spades given that Card1 is QH?

[Van den Broeck; AAAI-KRR’15]
A Simple Reasoning Problem

Probability that Card52 is Spades given that Card1 is QH?

13/51

[Van den Broeck; AAAI-KRR'15]
Let us automate this:

1. CNF encoding for deck of cards
2. Compile to tractable knowledge base (e.g., d-DNNF)
3. Condition on observations/questions
   “Card1 is hearts”
4. Model counting
Automated Reasoning

Let us automate this:

1. CNF encoding for deck of cards
2. Compile to tractable knowledge base (e.g., d-DNNF)
3. Condition on observations/questions
   "Card1 is hearts"
4. Model counting

A typical BeyondNP pipeline!
Automated Reasoning

Let us automate this:

1. **CNF encoding for deck of cards**

   Card(p1,c1) v Card(p1,c2) v ...
   Card(p1,c1) v Card(p2,c1) v ...
   ¬Card(p1,c1) v ¬Card(p1,c2)
   ¬Card(p1,c2) v ¬Card(p1,c3)
   ...
   ¬Card(p2,c1) v ¬Card(p2,c2)
   ...

Let us automate this:

1. CNF encoding for deck of cards
2. Compile to tractable knowledge base (e.g., d-DNNF)
3. Condition on observations/questions
   "Card1 is hearts"
4. Model counting

Which language to choose?

Cards problem is easy: we want to be polynomial.
Deck of Cards Graphically

- A♥
- 2♥
- 3♥
- ...
- K♥

Card(K♥, p14)

2. Compile to tractable knowledge base
3. Condition on observations/questions
4. Model counting
Deck of Cards Graphically

2. Compile to tractable knowledge base
3. Condition on observations/questions
4. Model counting
2. Compile to tractable knowledge base
3. **Condition on observations/questions**
4. Model counting

Card(K♥, p14)

¬ Card(K♥, p14)
2. Compile to tractable knowledge base
3. **Condition on observations/questions**
4. Model counting

\[ \neg \text{Card(K♥,p14)} \]
2. Compile to tractable knowledge base
3. Condition on observations/questions
4. **Model counting**
2. Compile to tractable knowledge base
3. Condition on observations/questions
4. Model counting: How many perfect matchings?
Deck of Cards Graphically

2. Compile to tractable knowledge base
3. Condition on observations/questions
4. **Model counting:** How many *perfect matchings*?
Observations

- Deck of cards = complete bigraph
- CD = removing edges in bigraph
  Encode any bigraph in cards problem
- CT = counting perfect matchings
- Problem is $\#P$-complete!

No language with CD and CT can represent the cards problem compactly, unless P=NP.
What's Going On Here?

Probability that Card52 is Spades given that Card1 is QH?

[Van den Broeck; AAAI-KRR’15]
What's Going On Here?

Probability that Card52 is Spades given that Card1 is QH? 13/51

[Van den Broeck; AAAI-KRR’15]
What's Going On Here?

Probability that Card52 is Spades given that Card1 is QH?

13/51

[Van den Broeck; AAAI-KRR’15]
What's Going On Here?

Probability that Card52 is Spades given that Card2 is QH?

[Van den Broeck; AAAI-KRR’15]
What's Going On Here?

Probability that Card52 is Spades given that Card2 is QH? 13/51

[Van den Broeck; AAAI-KRR’15]
What's Going On Here?

Probability that Card52 is Spades given that Card3 is QH?

[Van den Broeck; AAAI-KRR'15]
What's Going On Here?

Probability that Card52 is Spades given that Card3 is QH? 13/51

[Van den Broeck; AAAI-KRR’15]
Tractable Reasoning

What's going on here?
Which property makes reasoning tractable?

[Niepert, Van den Broeck; AAAI’14], [Van den Broeck; AAAI-KRR’15]
Tractable Reasoning

What's going on here? Which property makes reasoning tractable?

- High-level (first-order) reasoning
- Symmetry
- Exchangeability

⇒ Lifted Inference

[Niepert, Van den Broeck; AAAI’14], [Van den Broeck; AAAI-KRR’15]
Let us automate this:

- Relational/FO model

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

- First-Order Knowledge Compilation
MOTIVATION 2
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>

$\#\text{SAT} = 3$

[Valiant] #P-hard, even for 2CNF
Weighted Model Counting

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

<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>

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

$\#SAT = 3$
Weighted Model Counting

- Model = solution to a propositional logic formula $\Delta$
- Model counting = $\#\text{SAT}$
- Weighted model counting (WMC)
  - Weights for assignments to variables
  - Model weight is product of variable weights $w(.)$

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

<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>1 * 3 = 3</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 * 3 = 6</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>Yes</td>
<td>2 * 5 = 10</td>
</tr>
</tbody>
</table>

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

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

- Weighted model counting (WMC)
  - Weights for assignments to variables
  - Model weight is product of variable weights $w(.)$

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

<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>1 * 3 = 3</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>No</td>
<td>2 * 3 = 6</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>Yes</td>
<td>2 * 5 = 10</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>

$\#SAT = 3$  
$\text{WMC} = 19$
Assembly language for probabilistic reasoning and learning

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

Model = solution to first-order logic formula $\Delta$

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

Days = \{Monday\}
First-Order Model Counting

Model = solution to first-order logic formula $\Delta$

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

Days = {Monday}

<table>
<thead>
<tr>
<th>Rain(M)</th>
<th>Cloudy(M)</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>

$\text{FOMC} = 3$
Weighted First-Order Model Counting

Model = solution to **first-order** logic formula $\Delta$

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

<table>
<thead>
<tr>
<th>Days = {Monday, Tuesday}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rain(M)</td>
</tr>
<tr>
<td>T</td>
</tr>
<tr>
<td>T</td>
</tr>
<tr>
<td>F</td>
</tr>
<tr>
<td>F</td>
</tr>
<tr>
<td>T</td>
</tr>
<tr>
<td>T</td>
</tr>
<tr>
<td>F</td>
</tr>
<tr>
<td>F</td>
</tr>
<tr>
<td>T</td>
</tr>
<tr>
<td>T</td>
</tr>
<tr>
<td>F</td>
</tr>
<tr>
<td>F</td>
</tr>
<tr>
<td>T</td>
</tr>
<tr>
<td>T</td>
</tr>
<tr>
<td>F</td>
</tr>
<tr>
<td>F</td>
</tr>
</tbody>
</table>
### Weighted First-Order Model Counting

Model = solution to **first-order** logic formula $\Delta$

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

Days = \{Monday, Tuesday\}

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

$\text{#SAT} = 9$
Weighted First-Order Model Counting

Model = solution to first-order logic formula $\Delta$

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

Days = {Monday, Tuesday}

<table>
<thead>
<tr>
<th>Rain(M)</th>
<th>Cloudy(M)</th>
<th>Rain(T)</th>
<th>Cloudy(T)</th>
<th>Model?</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>Yes</td>
<td>$1 \times 1 \times 3 \times 3 = 9$</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>No</td>
<td>0</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>Yes</td>
<td>$2 \times 1 \times 3 \times 3 = 18$</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>Yes</td>
<td>$2 \times 1 \times 5 \times 3 = 30$</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>No</td>
<td>0</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>No</td>
<td>0</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>No</td>
<td>0</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>No</td>
<td>0</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>Yes</td>
<td>$1 \times 2 \times 3 \times 3 = 18$</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>No</td>
<td>0</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>Yes</td>
<td>$2 \times 2 \times 3 \times 3 = 36$</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>Yes</td>
<td>$2 \times 2 \times 5 \times 3 = 60$</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>Yes</td>
<td>$1 \times 2 \times 3 \times 5 = 30$</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>No</td>
<td>0</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>Yes</td>
<td>$2 \times 2 \times 3 \times 5 = 60$</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>Yes</td>
<td>$2 \times 2 \times 5 \times 5 = 100$</td>
</tr>
</tbody>
</table>

$\#\text{SAT} = 9$
### Weighted First-Order Model Counting

Model = solution to **first-order** logic formula $\Delta$

**Formula:**

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

**Days:**

$\{\text{Monday, Tuesday}\}$

**Weights:**

- $w(\text{R}) = 1$
- $w(\neg\text{R}) = 2$
- $w(\text{C}) = 3$
- $w(\neg\text{C}) = 5$

<table>
<thead>
<tr>
<th>Rain(M)</th>
<th>Cloudy(M)</th>
<th>Rain(T)</th>
<th>Cloudy(T)</th>
<th>Model?</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>Yes</td>
<td>$1 \times 1 \times 3 \times 3 = 9$</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>No</td>
<td>0</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>Yes</td>
<td>$2 \times 1 \times 3 \times 3 = 18$</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>Yes</td>
<td>$2 \times 1 \times 5 \times 3 = 30$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rain(M)</th>
<th>Cloudy(M)</th>
<th>Rain(T)</th>
<th>Cloudy(T)</th>
<th>Model?</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>No</td>
<td>0</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>No</td>
<td>0</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>No</td>
<td>0</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>No</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rain(M)</th>
<th>Cloudy(M)</th>
<th>Rain(T)</th>
<th>Cloudy(T)</th>
<th>Model?</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>Yes</td>
<td>$1 \times 2 \times 3 \times 3 = 18$</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>No</td>
<td>0</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>Yes</td>
<td>$2 \times 2 \times 3 \times 3 = 36$</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>Yes</td>
<td>$2 \times 2 \times 5 \times 3 = 60$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rain(M)</th>
<th>Cloudy(M)</th>
<th>Rain(T)</th>
<th>Cloudy(T)</th>
<th>Model?</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>Yes</td>
<td>$1 \times 2 \times 3 \times 5 = 30$</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>No</td>
<td>0</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>Yes</td>
<td>$2 \times 2 \times 3 \times 5 = 60$</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>Yes</td>
<td>$2 \times 2 \times 5 \times 5 = 100$</td>
</tr>
</tbody>
</table>

\[\text{WFOMC} = 361\]

\[\#\text{SAT} = 9\]
Assembly language for high-level probabilistic reasoning and learning

- Parfactor graphs
- Probabilistic logic programs
- Markov Logic
- Relational Bayesian networks
- Probabilistic databases

Weighted First-Order Model Counting

[VdB et al.; IJCAI’11, PhD’13, KR’14, UAI’14]
Statistical Relational Learning

An MLN = set of constraints \((w, \Gamma(x))\)

Weight of a world = product of \(w\), for all rules \((w, \Gamma(x))\) and groundings \(\Gamma(a)\) that hold in the world

\[
P_{MLN}(Q) = \frac{\text{[sum of weights of worlds of } Q]\} Z
\]

Applications: large probabilistic KBs
FO NNF SYNTAX
First-Order Knowledge Compilation

- Input: Sentence in FOL
- Output: Representation tractable for some class of queries.
- In this work:
  - Function-free FOL
  - Model counting in NNF tradition
- Some pre-KC-map work:
  - FO Horn clauses
  - FO BDDs
Alphabet

- **FOL**
  - Predicates/relations: Friends
  - Object names: \( x, y, z \)
  - Object variables: \( X, Y, Z \)
  - Symbols classical FOL (\( \forall, \exists, \land, \lor, \neg, \ldots \))

- **Group logic**
  - Group variables: \( X, Y, Z \)
  - Symbols from basic set theory
    (e.g., \( \cup, \cap, \in, \subseteq, \{, \}, \) complement).
Syntax

- Object terms: X, alice, bob
- Group terms: X, {alice, bob}, X ∪ Y
- Atom: Friends(alice, X)
- Formulas:
  - (α), ¬α, α ∨ β, and α ∧ β
  - ∀X ∈ G, α and ∃X ∈ G, α
  - ∀X ⊆ G, α and ∃X ⊆ G, α
- Group logic syntactic sugar:
  - P(G) is ∀X ∈ G, P(X)
  - ¬P(G) is ∀X ∈ G, ¬P(X)
Examples:

• \( \forall X \in G, Y \in \{ \text{alice, bob} \}, \)
  Enemies(X, Y )
  \( \Rightarrow \neg \text{Friends}(X, Y ) \land \neg \text{Friends}(Y, X) \)

• \( \forall X \in G, Y \in G, \)
  Smokes(X) \land \text{Friends}(X, Y ) \Rightarrow \text{Smokes}(Y) \)

• \( \exists G \subseteq \{ \text{alice, bob} \}, \text{Smokes}(G) \land \neg \text{Healthy}(G) \)
Semantics

• Template language for propositional logic
• Grounding a sentence: \( gr(\alpha) \)
  • Replace \( \forall \) by \( \land \)
  • Replace \( \exists \) by \( \lor \)
  • End result: ground sentence = propositional logic

• Grounding is polynomial in group sizes when no \( \forall X \subseteq G \) or \( \exists X \subseteq G \)

Important for polytime reduction to NNF circuits
Decomposability

• **Conjunction**: \( \alpha(X, G) \land \beta(X, G) \)
  
  For any substitution \( X = c \) and \( G = g \), we have that
  \( \text{gr}(\alpha(c, g)) \land \text{gr}(\beta(c, g)) \)
  is decomposable

  Meaning: \( \alpha \) and \( \beta \) can never talk about the same ground atoms

• **Quantifier**: \( \forall Y \in G, \alpha(Y) \)
  
  For any two \( a, b \in G \), we have that
  \( \text{gr}(\alpha(a)) \land \text{gr}(\alpha(b)) \)
  is decomposable
Determinism

- **Disjunction**: $\alpha(X, G) \lor \beta(X, G)$
  
  For any substitution $X=c$ and $G=g$, we have that $gr(\alpha(c,g)) \lor gr(\beta(c,g))$ is deterministic

  Meaning: $\alpha \land \beta$ is UNSAT

- **Quantifier**: $\exists Y \in G, \alpha(Y)$
  
  For any two $a,b \in G$, we have that $gr(\alpha(a)) \lor gr(\alpha(b))$ is decomposable
Group Quantifiers

- **Decomposability**: \( \forall X \subseteq G, \alpha(X) \)
  For any two \( A, B \subseteq G \), we have that \( \text{gr}(\alpha(A)) \lor \text{gr}(\alpha(B)) \) is decomposable

- **Determinism**: \( \exists X \subseteq G, \alpha(X) \)
  For any two \( A, B \subseteq G \), we have that \( \text{gr}(\alpha(A)) \lor \text{gr}(\alpha(B)) \) is deterministic
Automorphism

• Object permutation $\sigma : D \rightarrow D$ is a one-to-one mapping from objects to objects.
• Permuting $\alpha$ using $\sigma$ replaces $o$ in $\alpha$ by $\sigma(o)$.
• Sentences $\alpha$ and $\beta$ are $p$-equivalent iff $\alpha$ is equivalent to an object permutation of $\beta$.
  Smokes(alice) and Smokes(bob) are $p$-equivalent
• Group quantifiers: $\forall X \subseteq G, \alpha(X)$ or $\exists X \subseteq G, \alpha(X)$
  Are **automorphic** iff for any two $A, B \subseteq G$ s.t. $|A| = |B|$, $\text{gr}(\alpha(A))$ and $\text{gr}(\alpha(B))$ are $p$-equivalent
First-Order NNF

∀X, X ∈ People : belgian(X) \implies \text{likes}(X, \text{chocolate})
First-Order NNF

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

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

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

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

Deterministic
First-Order d-DNNF

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

∀X, X ∈ People : belgian(X) ⇒ likes(X, chocolate)
First-Order ad-DNNF

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

- FO NNF: group logic circuits, negation only on atoms
- FO d-DNNF: determinism and decomposability
  Grounding generates a d-DNNF
- FO DNNF
  Grounding generates a DNNF
- FO ad-DNNF: automorphic
  Powerful properties!
FO NNF TRACTABILITY
Symmetric WFOMC

**Def.** A weighted vocabulary is \((R, w)\), where

- \(R = (R_1, R_2, \ldots, R_k)\) = relational vocabulary
- \(w = (w_1, w_2, \ldots, w_k)\) = weights

- Fix an FO formula \(Q\), domain of size \(n\)
- The weight of a ground tuple \(t\) in \(R_i\) is \(w_i\)

Complexity of FOMC / WFOMC\((Q, n)\)?

Data/domain complexity:
- fixed \(Q\), input \(n\) / and \(w\)
Symmetric WFOMC on FO ad-DNNF

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

Complexity polynomial in domain size!
Polynomial in NNF size for bounded depth.
FOMC Query: Example

FO-Model Counting: $w(R) = w(\neg R) = 1$
FO ad-DNNF sentences
4. \[ \Delta = (\text{Stress(Alice)} \Rightarrow \text{Smokes(Alice)}) \]

Domain = \{Alice\}

FO-Model Counting: \( w(R) = w(\neg R) = 1 \)

FO ad-DNNF sentences

FOMC Query: Example
FO-MC Query: Example

FO-Model Counting: \( w(R) = w(\neg R) = 1 \)
FO ad-DNNF sentences

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

\( \rightarrow 3 \) models

Domain = \{Alice\}
FO-Model Counting: $w(R) = w(\neg R) = 1$

FO ad-DNNF sentences

4. $\Delta = (\text{Stress}(\text{Alice}) \Rightarrow \text{Smokes(\text{Alice})})$
   \[ \Delta = \forall x, (\text{Stress}(x) \Rightarrow \text{Smokes}(x)) \]
   \[ \rightarrow 3 \text{ models} \]
   \[ \text{Domain} = \{\text{Alice}\} \]

3. $\Delta = \forall x, (\text{Stress}(x) \Rightarrow \text{Smokes}(x))$
   \[ \text{Domain} = \{n \text{ people}\} \]
FOMC Query: Example

FO-Model Counting: \( w(R) = w(\neg R) = 1 \)

FO ad-DNNF sentences

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

\( \rightarrow 3 \) models

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

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

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

Domain = \{n people\}
FOMC Query: Example

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

2. \( \Delta = \forall y, (\text{ParentOf}(y) \land \text{Female} \implies \text{MotherOf}(y)) \)  
   D = \{n people\}
FOMC Query: Example

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

2. \( \Delta = \forall y, (\text{ParentOf}(y) \land \text{Female} \Rightarrow \text{MotherOf}(y)) \)
   
   D = \{n \text{ people}\}

If Female = true? \( \Delta = \forall y, (\text{ParentOf}(y) \Rightarrow \text{MotherOf}(y)) \)

\( \Rightarrow 3^n \text{ models} \)
FOMC Query: Example

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

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

If Female = false? \[ \Delta = \text{true} \]
   \[ \rightarrow 4^n \text{ models} \]
FOMC Query: Example

2. \[ \Delta = \forall y, \text{(ParentOf}(y) \land \text{Female} \Rightarrow \text{MotherOf}(y)) \]
   \[ D = \{n \text{ people}\} \]
   If Female = true? \[ \Delta = \forall y, \text{(ParentOf}(y) \Rightarrow \text{MotherOf}(y)) \] \[ \rightarrow 3^n \text{ models} \]
   If Female = false? \[ \Delta = \text{true} \] \[ \rightarrow 4^n \text{ models} \]
   \[ \rightarrow 3^n + 4^n \text{ models} \]

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

<table>
<thead>
<tr>
<th></th>
<th>Query</th>
<th>Models</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>$\Delta = \forall x, \forall y, (\text{ParentOf}(x,y) \land \text{Female}(x) \Rightarrow \text{MotherOf}(x,y))$</td>
<td>$3^n$</td>
</tr>
<tr>
<td></td>
<td>$D = {n \text{ people}}$</td>
<td></td>
</tr>
<tr>
<td>2.</td>
<td>$\Delta = \forall y, (\text{ParentOf}(y) \land \text{Female} \Rightarrow \text{MotherOf}(y))$</td>
<td>$3^n$</td>
</tr>
<tr>
<td></td>
<td>$D = {n \text{ people}}$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>If Female = true?</td>
<td>$\Delta = \forall y, (\text{ParentOf}(y) \Rightarrow \text{MotherOf}(y))$</td>
</tr>
<tr>
<td></td>
<td>If Female = false?</td>
<td>$\Delta = \text{true}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$3^n + 4^n$ models</td>
</tr>
<tr>
<td>3.</td>
<td>$\Delta = \forall x, (\text{Stress}(x) \Rightarrow \text{Smokes}(x))$</td>
<td>$3^n$</td>
</tr>
<tr>
<td></td>
<td>$\text{Domain} = {n \text{ people}}$</td>
<td></td>
</tr>
</tbody>
</table>
## FOMC Query: Example

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1.</strong></td>
<td>( \Delta = \forall x, \forall y, (\text{ParentOf}(x,y) \land \text{Female}(x) \Rightarrow \text{MotherOf}(x,y)) )</td>
<td>( \text{D} = {\text{n people}} )</td>
</tr>
<tr>
<td></td>
<td>( \rightarrow (3^n + 4^n)^n \text{ models} )</td>
<td></td>
</tr>
<tr>
<td><strong>2.</strong></td>
<td>( \Delta = \forall y, (\text{ParentOf}(y) \land \text{Female} \Rightarrow \text{MotherOf}(y)) )</td>
<td>( \text{D} = {\text{n people}} )</td>
</tr>
<tr>
<td>If Female = true?</td>
<td>( \Delta = \forall y, (\text{ParentOf}(y) \Rightarrow \text{MotherOf}(y)) )</td>
<td>( \rightarrow 3^n \text{ models} )</td>
</tr>
<tr>
<td>If Female = false?</td>
<td>( \Delta = \text{true} )</td>
<td>( \rightarrow 4^n \text{ models} )</td>
</tr>
<tr>
<td></td>
<td>( \rightarrow 3^n + 4^n \text{ models} )</td>
<td></td>
</tr>
<tr>
<td><strong>3.</strong></td>
<td>( \Delta = \forall x, (\text{Stress}(x) \Rightarrow \text{Smokes}(x)) )</td>
<td>( \text{Domain} = {\text{n people}} )</td>
</tr>
<tr>
<td></td>
<td>( \rightarrow 3^n \text{ models} )</td>
<td></td>
</tr>
</tbody>
</table>
Group Quantifiers: Example

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

• Not decomposable!
• Rewrite as FO ad-DNNF:

\[ \exists G \subseteq D, \text{Smokes}(G) \land \overline{\text{Smokes}}(\overline{G}) \land \overline{\text{Friends}}(G, \overline{G}) \]

• Not possible to ground to d-DNNF!
• How to do tractable CT?

\[ \sum_{i=0}^{\left| \tau \right|} \left( \frac{\left| \tau \right|}{i} \right) \cdot U(\beta(X/x_i)) \quad \text{when } \alpha = \exists X \subseteq \tau, \beta, \text{ and } x_i \text{ is any subset of } \tau \text{ such that } |x_i| = i \]
Group Quantifiers: Example

\[ \exists G \subseteq D, \text{Smokes}(G) \land \neg \text{Smokes}(\bar{G}) \land \text{Friends}(G, \bar{G}) \]
Group Quantifiers: Example

$\exists G \subseteq D, \text{Smokes}(G) \land \text{Smokes}(G) \land \text{Friends}(G, \bar{G})$

- If we know $G$ 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>
Group Quantifiers: Example

\[ \exists G \subseteq D, \text{Smokes}(G) \land \text{Smokes}(\bar{G}) \land \text{Friends}(G, \bar{G}) \]

- If we know \( G \) 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 showing the relationship between Smokes, Friends, and Smokes]
Group Quantifiers: Example

\[ \exists G \subseteq D, \text{Smokes}(G) \land \overline{\text{Smokes}}(\bar{G}) \land \text{Friends}(G, \bar{G}) \]

- If we know \( G \) 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 showing the concept of group quantifiers with a database example.](image)
Group Quantifiers: Example

\[ \exists G \subseteq D, \text{Smokes}(G) \land \text{Smokes}(\overline{G}) \land \text{Friends}(G, \overline{G}) \]

- If we know \( G \) 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>Smoke</th>
<th>Friends</th>
<th>Smoke</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>
```
Group Quantifiers: Example

\[ \exists G \subseteq D, \text{Smokes}(G) \land \overline{\text{Smokes}}(\overline{G}) \land \text{Friends}(G, \overline{G}) \]

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

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

![Diagram showing group quantifiers with \( k \) smokers and \( n-k \) non-smokers connected through the friends relation.]
Group Quantifiers: Example

\[ \exists G \subseteq D, \text{Smokes}(G) \land \neg \text{Smokes}(\bar{G}) \land \text{Friends}(G, \bar{G}) \]

- If we know \( G \) 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 showing the relationship between Smokes, Friends, and Smokes with nodes labeled k and n-k for smokers and non-smokers, respectively.]
Group Quantifiers: Example

\[ \exists G \subseteq D, \text{Smokes}(G) \land \text{Smokes}(\bar{G}) \land \text{Friends}(G, \bar{G}) \]

- If we know $G$ 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 showing relationships between Smokes, Friends, and Smokes.]
Group Quantifiers: Example

\[ \exists G \subseteq D, \text{Smokes}(G) \land \text{Smokes}(\bar{G}) \land \text{Friends}(G, \bar{G}) \]

- If we know \( G \) 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 showing relationships between Smokes, Friends, and Smokes]
Group Quantifiers: Example

\[ \exists G \subseteq D, \text{Smokes}(G) \land \neg\text{Smokes}(\neg G) \land \text{Friends}(G, \neg G) \]

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

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

\[ \begin{array}{c}
\text{Smokes} \\
\text{Friends} \\
\text{Smokes}
\end{array} \]

- k
- n-k
- k
- n-k
Group Quantifiers: Example

\[ \exists G \subseteq D, \text{Smokes}(G) \land \neg \text{Smokes}(G) \land \text{Friends}(G, G) \]

- If we know \( G \) 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
Group Quantifiers: Example

\( \exists G \subseteq D, \text{Smokes}(G) \land \neg\text{Smokes}(\neg G) \land \text{Friends}(G, \neg G) \)

- If we know \( G \) 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?
Group Quantifiers: Example

\[ \exists G \subseteq D, \text{Smokes}(G) \land \text{Smokes}(\overline{G}) \land \text{Friends}(G, \overline{G}) \]

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

  **Database:**
  
  \[
  \begin{align*}
  \text{Smokes}\text{(Alice)} &= 1 \\
  \text{Smokes}\text{(Bob)} &= 0 \\
  \text{Smokes}\text{(Charlie)} &= 0 \\
  \text{Smokes}\text{(Dave)} &= 1 \\
  \text{Smokes}\text{(Eve)} &= 0 \\
  \ldots
  \end{align*}
  \]

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

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

  \[ \rightarrow \binom{n}{k} 2^{n^2 - k(n-k)} \text{ models} \]
Group Quantifiers: Example

\[ \exists G \subseteq D, \text{Smokes}(G) \land \neg\text{Smokes}(\bar{G}) \land \text{Friends}(G, \bar{G}) \]

- If we know \( G \) 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

- In total...
Group Quantifiers: Example

\[
\exists G \subseteq D, \text{Smokes}(G) \land \neg \text{Smokes}(\neg G) \land \text{Friends}(G, \neg G)
\]

- If we know \(G\) 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
Playing Cards Revisited

Let us automate this:

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

Playing Cards Revisited

Let us automate this:

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

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

Playing Cards Revisited

Let us automate this:

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

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

Computed in time polynomial in n

FO COMPILATION
Compilation Rules

• Lots of preprocessing
• Shannon decomposition/Boole’s expansion
• Detect propositional decomposability
• FO Shannon decomposition:
  \[ \exists X \subseteq \tau, P(X) \land \overline{P(X)} \land \beta \]
  Simplify \( \beta \) (remove atoms subsumed by \( P(X) \))
  Always deterministic! Ensure automorphic \( \exists \)
• Detect FO decomposability
FO NNF EXPRESSIVENESS
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 $\text{FO}$ ad-DNNF is complete for $\text{FO}^2$
- Model counting for $\text{FO}^2$ in PTIME domain complexity
Main Negative Results

Domain complexity:

- There exists an FO formula $Q$ s.t. symmetric FOMC($Q, n$) is $\#P_1$ hard
- There exists $Q$ in FO$^3$ s.t. FOMC($Q, n$) is $\#P_1$ hard
- There exists a conjunctive query $Q$ s.t. symmetric WFOMC($Q, n$) is $\#P_1$ hard
- There exists a positive clause $Q$ w.o. ‘$=$’ s.t. symmetric WFOMC($Q, n$) is $\#P_1$ hard

Therefore, no FO ad-DNNF can exist 😞
Proof

**Theorem.** There exists an FO³ sentence $Q$ s.t. $\text{FOMC}(Q,n)$ is $\#P_1$-hard

**Proof**

- **Step 1.** Construct a Turing Machine $U$ s.t.
  - $U$ is in $\#P_1$ and runs in linear time in $n$
  - $U$ computes a $\#P_1$-hard function

- **Step 2.** Construct an FO³ sentence $Q$ s.t. $\text{FOMC}(Q,n) / n! = U(n)$
[VdB; NIPS’11], [VdB et al.; KR’14], [Gribkoff, VdB, Suciu; UAI’15], [Beame, VdB, Gribkoff, Suciu; PODS’15], etc.
Fertile Ground

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

[VD; NIPS’11], [VD et al.; KR’14], [Gribkoff, VdB, Suciu; UAI’15], [Beame, VdB, Gribkoff, Suciu; PODS’15], etc.
Other Queries and Transformations

• What if all ground atoms have different weights? Asymmetric WFOMC
• FO d-DNNF complete for all monotone FO CNFs that support efficient CT
• No clausal entailment
• No conditioning
Conclusions

• Very powerful already!
• We need to solve this!

THANKS
References

- **Cards Example:**

- **First-Order Knowledge Compilation:**

- **Expressiveness:**