Lifted Probabilistic Inference by First-Order Knowledge Compilation

Guy Van den Broeck
Nima Taghipour
Wannes Meert
Jesse Davis
Luc De Raedt

Lifted Inference in Probabilistic Logical Models - Tutorial - IJCAI11
18/07/11
Outline

- Overview Approach
- First-Order d-DNNF Circuits
- First-Order Knowledge Compilation
- Experiments
- Conclusions
Outline

- Overview Approach
- First-Order d-DNNF Circuits
- First-Order Knowledge Compilation
- Experiments
- Conclusions
## Context

<table>
<thead>
<tr>
<th></th>
<th>Variable Elimination</th>
<th>Belief Propagation</th>
<th>KnowledgeCompilation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ground</td>
<td>[Zhang94]</td>
<td>[Pearl82]</td>
<td>[Darwiche03]</td>
</tr>
<tr>
<td>Lifted</td>
<td>[Poole03]</td>
<td>[Singla08]</td>
<td><strong>Our approach</strong></td>
</tr>
</tbody>
</table>
Advantages of Knowledge Compilation

- Compile once, then run polytime inference for multiple queries and evidence
- Efficient data structures
- Principled logical approach
- Exploits context-specific independences
- State of the art for exact inference in
  - Bayesian networks
  - Statistical relational learning
- Used in many domains, not just probabilistic reasoning
Can we lift knowledge compilation to a first-order setting?

First step taken: first-order d-DNNFs for
- weighted first-order model counting
- lifted probabilistic inference

Many open questions remaining!
What is Lifted Inference?

\[ 2\ \text{friends}(X, Y) \land \text{smokes}(X) \implies \text{smokes}(Y) \]
What is Lifted Inference?

2 \text{ friends}(X, Y) \land \text{smokes}(X) \Rightarrow \text{smokes}(Y)

- Variables $X, Y$ range over domain People
- Represents propositional model for given domain (50 people)
- Propositional inference in factor graph is expensive
- However: symmetries
What is Lifted Inference?

2 friends(X, Y) ∧ smokes(X) ⇒ smokes(Y)

- We compile to a circuit independent of |People|
- Inference linear in |People|

→ Lifted Inference
Knowledge Compilation

- Bayesian Network
- Factor Graph
- MLN
- ...

First-Order Knowledge Compilation
Knowledge Compilation

• Step 1: Convert model to weighted CNF

- Bayesian Network
- Factor Graph
- MLN
- ...
Knowledge Compilation

- Step 1: Convert model to weighted CNF
- Step 2: Convert CNF to d-DNNF circuit
Knowledge Compilation

- **Step ①**: Convert model to weighted CNF
- **Step ②**: Convert CNF to d-DNNF circuit
- **Step ③**: Perform weighted model counting
Our Approach: First-Order Knowledge Compilation:

- Step 1: Convert model to weighted FO CNF
- Step 2: Convert CNF to FO d-DNNF circuit
- Step 3: Perform weighted FO model counting
Step 1: Converting to Weighted FO CNF

MLN

\[ w : \text{smokes}(X) \land \text{friends}(X, Y) \Rightarrow \text{smokes}(Y) \]
Step 1: Converting to Weighted FO CNF

MLN

Weighted FO Theory

\[ w : \text{smokes}(X) \land \text{friends}(X, Y) \Rightarrow \text{smokes}(Y) \]

\[ [\text{smokes}(X) \land \text{friends}(X, Y) \Rightarrow \text{smokes}(Y)] \equiv f(X, Y) \]
Step 1: Converting to Weighted FO CNF

MLN

Weighted FO Theory

Weighted FO CNF

\[ w : \text{smokes}(X) \land \text{friends}(X, Y) \Rightarrow \text{smokes}(Y) \]

\[ [\text{smokes}(X) \land \text{friends}(X, Y) \Rightarrow \text{smokes}(Y)] \equiv f(X, Y) \]

\[ \text{smokes}(Y) \lor \neg \text{smokes}(X) \lor \neg \text{friends}(X, Y) \lor \neg f(X, Y) \]

\[ \text{friends}(X, Y) \lor f(X, Y) \]

\[ \text{smokes}(X) \lor f(X, Y) \]

\[ \neg \text{smokes}(Y) \lor f(X, Y) \]
Step 3: Weighted FO Model Counting

- Weight function on ground atoms

\[
\begin{align*}
    w(\text{smokes}(X)) &= 2 & w(\neg \text{smokes}(X)) &= 10 \\
    w(\text{friends}(X, Y)) &= 5 & w(\neg \text{friends}(X, Y)) &= 1
\end{align*}
\]
Step 3: Weighted FO Model Counting

- Weight function on ground atoms
  \[
  w(\text{smokes}(X)) = 2 \quad w(\neg\text{smokes}(X)) = 10
  \]
  \[
  w(\text{friends}(X, Y)) = 5 \quad w(\neg\text{friends}(X, Y)) = 1
  \]

- Weight of a model (possible world)
  \[
  \{\text{smokes}(alice), \neg\text{smokes}(bob), \text{friends}(alice, bob), \ldots\}
  \]
  \[
  2 \cdot 10 \cdot 5 \cdots = 100
  \]
Step 3: Weighted FO Model Counting

- **Weight function on ground atoms**
  \[
  \begin{align*}
  w(\text{smokes}(X)) &= 2 \\
  w(\neg\text{smokes}(X)) &= 10 \\
  w(\text{friends}(X, Y)) &= 5 \\
  w(\neg\text{friends}(X, Y)) &= 1
  \end{align*}
  \]

- **Weight of a model (possible world)**
  \[
  \{\text{smokes}(alice), \neg\text{smokes}(bob), \text{friends}(alice, bob), \ldots\} \]
  \[
  2 \cdot 10 \cdot 5 \cdot \ldots = 100
  \]

- **Weight of all models is**
  \[
  100 + \ldots = Z
  \]
Step 3: Weighted FO Model Counting

- Weight function on ground atoms
  
  \[ w(\text{smokes}(X)) = 2 \quad w(\neg \text{smokes}(X)) = 10 \]
  
  \[ w(\text{friends}(X, Y)) = 5 \quad w(\neg \text{friends}(X, Y)) = 1 \]

- Weight of a model (possible world)
  
  \[ \{ \text{smokes}(alice), \neg \text{smokes}(bob), \text{friends}(alice, bob), \ldots \} \]

  \[ 2 \cdot 10 \cdot 5 \cdot \ldots = 100 \]

- Weight of all models is \( 100 + \ldots = Z \)

  Weight of models where Alice smokes is \( 100 + \ldots = Q \)
Step 3: Weighted FO Model Counting

- Weight function on ground atoms
  \[ w(\text{smokes}(X)) = 2 \quad w(\neg \text{smokes}(X)) = 10 \]
  \[ w(\text{friends}(X, Y)) = 5 \quad w(\neg \text{friends}(X, Y)) = 1 \]

- Weight of a model (possible world)
  \[ \{ \text{smokes}(alice), \neg \text{smokes}(bob), \text{friends}(alice, bob), \ldots \} \]
  \[ 2 \cdot 10 \cdot 5 \cdot \ldots = 100 \]

- Weight of all models is \( 100 + \ldots = Z \)

Weight of models where Alice smokes is \( 100 + \ldots = Q \)

- \[ P(\text{smokes}(alice)) = \frac{Q}{Z} \]
Outline

- Overview Approach
- First-Order d-DNNF Circuits
- First-Order Knowledge Compilation
- Experiments
- Conclusions
Logical theory:

\[ p \lor r \]

\[ q \lor \neg r \]
Propositional d-DNNF Circuits

[Darwiche and Marquis, 2002]

Logical theory:

\[ p \lor r \]

\[ q \lor \neg r \]
Propositional $d$-DNNF Circuits

[Darwiche and Marquis, 2002]

Logical theory:

\[ p \lor r \]
\[ q \lor \neg r \]

Logical operators (inner node)

Literal (leaf)
Propositional d-DNNF Circuits

[Darwiche and Marquis, 2002]

Logical theory:

\[ p \lor r \]
\[ q \lor \neg r \]

Logical operators (inner node)

Deterministic disjunction

Literal (leaf)
Propositional d-DNNF Circuits

[Darwiche and Marquis, 2002]

Logical theory:

\[ p \lor r \]
\[ q \lor \neg r \]

Logical operators (inner node):

Deterministic disjunction

Decomposable conjunction

Literal (leaf)
First-Order $d$-DNNF Circuits

Logical Theory:
\[
p(X) \lor r
\]
\[
q(X) \lor \neg r
\]
First-Order d-DNNF Circuits

Logical Theory:

\[ p(X) \lor r \]
\[ q(X) \lor \neg r \]

First-Order Literal (leaf)
First-Order d-DNNF Circuits

Logical Theory:

\[ p(X) \lor r \]

\[ q(X) \lor \neg r \]

- Deterministic disjunction
- Decomposable conjunction
- 3 additional first-order operators (inner nodes)
Outline

- Overview Approach
- First-Order d-DNNF Circuits
- First-Order Knowledge Compilation
- Experiments
- Conclusions
Step 2: Our Compilation Algorithm

- Recursively apply
  - Unit Propagation
  - Independence
  - Inclusion-Exclusion (Shannon Decomposition)
  - Shattering
  - Independent Partial Grounding
  - Atom Counting
  - (Grounding)
Step 2: Our Compilation Algorithm

- Recursively apply
  - Unit Propagation
  - Independence
  - Inclusion-Exclusion (Shannon Decomposition)
  - Shattering
  - Independent Partial Grounding
  - Atom Counting
  - (Grounding)

Generate **first-order operators** in inner nodes

- Weighted FO CNF
- FO d-DNNF Circuit
Step 2: Our Compilation Algorithm

- Recursively apply
  - Unit Propagation
  - Independence
  - Inclusion-Exclusion (Shannon Decomposition)
  - Shattering
  - Independent Partial Grounding
  - Atom Counting
  - (Grounding)
Unit Propagation

\[
\begin{align*}
\text{friends}(X, Y) \lor \text{dislikes}(X, Y) \\
\neg \text{friends}(X, Y) \lor \text{likes}(X, Y) \\
\text{friends}(X, X)
\end{align*}
\]
Unit Propagation

Unit clause

\[ \text{friends}(X, Y) \lor \text{dislikes}(X, Y) \]
\[ \neg \text{friends}(X, Y) \lor \text{likes}(X, Y) \]
\[ \text{friends}(X, X) \]
Unit Propagation

friends(\(X, Y\)) \lor \text{dislikes}(\(X, Y\))
\neg \text{friends}(\(X, Y\)) \lor \text{likes}(\(X, Y\))
friends(\(X, X\))
Unit Propagation

Clauses split w.r.t. unit clause atom 'residuals' → independent

friends(X, Y) ∨ dislikes(X, Y)
¬friends(X, Y) ∨ likes(X, Y)
friends(X, X)
Unit Propagation

Resolvent of unit and 2\textsuperscript{nd} clause

\begin{align*}
\text{friends}(X, Y) \lor \text{dislikes}(X, Y) \\
\neg \text{friends}(X, Y) \lor \text{likes}(X, Y) \\
\text{friends}(X, X)
\end{align*}
Step 2: Our Compilation Algorithm

- Recursively apply
  - Unit Propagation
  - Independence
  - Inclusion-Exclusion (Shannon Decomposition)
  - Shattering
  - Independent Partial Grounding
  - Atom Counting
  - (Grounding)

- Weighted FO CNF
- FO d-DNNF Circuit
Atom Counting

\[
\begin{align*}
\text{fun}(X) \lor \neg \text{friends}(X, Y) \\
\text{fun}(X) \lor \neg \text{friends}(Y, X)
\end{align*}
\]
Atom Counting

Atom with one logical variable \( X \in \{\text{luc, jesse}\} \)

\[
\begin{align*}
\text{fun}(X) \lor \neg \text{friends}(X, Y) \\
\text{fun}(X) \lor \neg \text{friends}(Y, X)
\end{align*}
\]
Atom Counting

Atom with 1 logical variable

\[
\begin{align*}
\text{fun}(X) \lor \neg \text{friends}(X, Y) \\
\text{fun}(X) \lor \neg \text{friends}(Y, X)
\end{align*}
\]
Atom Counting

fun(\(X\)) \lor \neg \text{friends}(X, Y)
fun(\(X\)) \lor \neg \text{friends}(Y, X)

All partial interpretations for fun(\(X\))
- deterministic
- \(2^{\text{People}}\)
Atom Counting

fun(X) \lor \neg \text{friends}(X, Y)
fun(X) \lor \neg \text{friends}(Y, X)

fun(jesse)
fun(luc)

fun(X) \lor \neg \text{friends}(X, Y)
fun(X) \lor \neg \text{friends}(Y, X)

\neg \text{fun}(jesse)
fun(luc)

fun(X) \lor \neg \text{friends}(X, Y)
fun(X) \lor \neg \text{friends}(Y, X)

\neg \text{fun}(jesse)
\neg \text{fun}(luc)

fun(X) \lor \neg \text{friends}(X, Y)
fun(X) \lor \neg \text{friends}(Y, X)

Same weighted model count
Atom Counting

\[
\begin{align*}
\text{fun}(X) \vee \neg \text{friends}(X, Y) \\
\text{fun}(X) \vee \neg \text{friends}(Y, X)
\end{align*}
\]

\[
2^{|\text{People}|} \rightarrow |\text{People}| + 1
\]
Atom Counting

fun(X) ∨ ¬friends(X, Y)
fun(X) ∨ ¬friends(Y, X)

Isomorphic circuits

fun(X), X ∈ People^T
¬fun(X), X ∈ People⊥
|People^T| = 2
|People⊥| = 0
fun(X) ∨ ¬friends(X, Y)
fun(X) ∨ ¬friends(Y, X)

fun(X), X ∈ People^T
¬fun(X), X ∈ People⊥
|People^T| = 1
|People⊥| = 1
fun(X) ∨ ¬friends(X, Y)
fun(X) ∨ ¬friends(Y, X)

fun(X), X ∈ People^T
¬fun(X), X ∈ People⊥
|People^T| = 0
|People⊥| = 2
fun(X) ∨ ¬friends(X, Y)
fun(X) ∨ ¬friends(Y, X)
Atom Counting

\[
\text{fun}(X) \lor \neg \text{friends}(X, Y) \\
\text{fun}(X) \lor \neg \text{friends}(Y, X)
\]

\[
\bigvee_{\text{People}^\top \subseteq \text{People}}
\]

\[
\text{fun}(X), X \in \text{People}^\top \\
\neg \text{fun}(X), X \in \text{People}^\bot \\
\text{fun}(X) \lor \neg \text{friends}(X, Y) \\
\text{fun}(X) \lor \neg \text{friends}(Y, X)
\]

\[|\text{People}|+1 \rightarrow 1\]
Atom Counting

fun(X) \lor \neg \text{friends}(X, Y)
fun(X) \lor \neg \text{friends}(Y, X)

\bigvee
FunPeople \subseteq People

\land

\land
fun(X), X \in FunPeople

\land
\neg \text{fun}(X), X \notin FunPeople

\land
\neg \text{friends}(X, Y), X \notin FunPeople

\land
\neg \text{friends}(X, Y), X \in FunPeople, Y \notin FunPeople
Outline

- Overview Approach
- First-Order d-DNNF Circuits
- First-Order Knowledge Compilation
- Experiments
- Conclusions
Evaluated Models

- Sick Death [de Salvo Braz 2005]
- WebKB [Lowd 2007]
- Competing Workshops [Milch 2008]
- Workshop Attributes [Milch 2008]
- Friends Smoker [Singla 2008]
- Friends Smoker Drinker
Probabilistic Model:

\(-2 : \text{hot}(W) \land \text{attends}(P)\)

\(3 : \text{attends}(P) \land \text{series}\)
Competing Workshops [Milch 2008]
Probabilistic Model:

1.2 : \text{smokes}(X) \land \text{friends}(X, Y) \Rightarrow \text{smokes}(Y)

1.2 : \text{smokes}(X) \land \text{friends}(Y, X) \Rightarrow \text{smokes}(Y)

2 : \text{smokes}(X) \Rightarrow \text{cancer}(X)
Friends Smoker [Singla 2008]
New Probabilistic Model:

1.2 : \textit{smokes}(X) \land \textit{friends}(X, Y) \Rightarrow \textit{smokes}(Y)

1.2 : \textit{drinks}(X) \land \textit{friends}(X, Y) \Rightarrow \textit{drinks}(Y)
Outline

- Overview Approach
- First-Order d-DNNF Circuits
- First-Order Knowledge Compilation
- Experiments
- Conclusions
Benefits of **First-Order Knowledge Compilation**?

- **Compile once** for a given set of evidence then run polytime inference
- **Efficient** data structure
- Principled **logical** approach
  - First **model theoretic** approach to lifted probabilistic inference
  - Uses concepts from **logical inference**: model counting, unit propagation, Shannon decomposition, etc.
- Exploits **context-specific independences**
- **State of the art** for exact lifted inference
  - Lifts more models than C-FOVE
Contributions

- We introduced first-order ...
  - knowledge compilation
  - d-DNNF circuits
  - weighted model counting
  - smoothing
- Algorithm to compile a first-order probabilistic model into FO d-DNNF circuits
- Closer to understanding the connection between lifted inference in first-order logic (resolution) and lifted inference in graphical models
Advertisement

• Poster
  Wednesday 10:30 UAI session

• Talk
  Thursday 10:30 UAI session
Thanks
Extra Slides
Logic-based Probabilistic Inference

- Bayesian Network
- Factor Graph
- MLN
- ProbLog

Weighted Propositional CNF

Weighted Model Counting

- DPLL Search Weighted Model Counting [Sang 2005]
- Knowledge Compilation [Darwiche]
Additional Operator Nodes

- Deterministic set-disjunction
- Decomposable set-conjunction
- Inclusion-Exclusion (non-deterministic disjunction)
### Auxiliary Operations

- **Splitting w.r.t. an atom** [Poole 2003]:
  - Similar to splitting in (C-)FOVE, but with domain constraints

<table>
<thead>
<tr>
<th>Before Splitting</th>
<th>Atom</th>
<th>After Splitting</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p(X, Y) \lor q(X) \lor r(Y) )</td>
<td>( p(X, X) )</td>
<td>( p(X, X) \lor q(X) \lor r(X) )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( p(X, Y) \lor q(X) \lor r(Y), X \neq Y )</td>
</tr>
<tr>
<td>( p(X) \lor q(X), X \in D )</td>
<td>( p(X), X \in D_1 )</td>
<td>( p(X) \lor q(X), X \in D_1 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( p(X) \lor q(X), X \in D_2 )</td>
</tr>
</tbody>
</table>

- **Shattering** [de Salvo Braz 2005]

  Splitting w.r.t. any atom in theory, until convergence
Unit Propagation

\[
q(X), X \in D_1^T \\
\neg q(X), X \in D_1^\perp \\
p(X,Y) \lor q(X) \\
p(X,Y) \lor r(Y)
\]

Unit propagation of \( q(X), X \in D_1^T \).
Independence

- Set of clauses independent from rest
- Independence when no unifying atoms

\[
\begin{align*}
p(X, Y) \lor q(X), X \neq a \\
p(a, X) \lor \neg r(X)
\end{align*}
\]
Inclusion-Exclusion

- Clause has set of literals that share no logical variables with rest
- Non-deterministic disjunction & intersection

\[ p(X) \lor r(X, X) \lor q(Y) \lor r(Y, Z) \]
Inclusion-Exclusion

- Clause has set of literals that share no logical variables with rest
- Non-deterministic disjunction & intersection

\[ p(X) \lor r(X, X) \lor q(Y) \lor r(Y, Z) \]

Inclusion-exclusion on \( p(X) \lor r(X, X) \lor q(Y) \lor r(Y, Z) \).

\[
\begin{align*}
&+ + - \\
p(X) \lor r(X, X) & q(X) \lor r(X, Y) & p(X) \lor r(X, X) \\
& q(X) \lor r(X, Y) & q(X) \lor r(X, Y)
\end{align*}
\]
Independent Partial Grounding

- Single logical variable in every atom (position!)
- Different partial groundings are independent

\[
p(X) \lor q(X, Y) \\
r(X) \lor q(X, Y)
\]
Independent Partial Grounding

- Single logical variable in every atom (position!)
- Different partial groundings are independent

\[
p(X) \lor q(X,Y) \\
r(X) \lor q(X,Y)
\]

Independent partial grounding of \( X \in D \).

\[
\bigwedge_{x \in D} \\
p(x) \lor q(x,X) \\
r(x) \lor q(x,X)
\]
Special Inclusion Exclusion Case: Shannon Decomposition

- CNF has ground atom
- Ground atoms do not share logical variables
- We can always add clause $r \lor \neg r$
- Intersection is unsatisfiable
- Inclusion-Exclusion becomes deterministic disjunction
First-Order Smoothing

FO d-DNNF

\[
p(X) \lor r
\]

\[
q(X) \lor \neg r
\]
First-Order Smoothing

**FO d-DNNF**

1. \( r \)
2. \( q(X), X \in D \)
3. \( \neg r \)
4. \( p(X), X \in D \)

\[
\begin{align*}
p(X) \lor r \\
q(X) \lor \neg r
\end{align*}
\]

**Smooth FO d-DNNF**

1. \( r \)
2. \( q(X), X \in D \)
3. \( \neg r \)
4. \( p(X), X \in D \)

\[
\begin{align*}
p(X) \lor \neg p(X), X \in D \\
q(X) \lor \neg q(X), X \in D
\end{align*}
\]
First-Order Smoothing

Complicated rules for
- atom counting
- independent partial groundings

FO d-DNNF

\[
\begin{align*}
\top & \rightarrow \land \\
\land & \rightarrow \land \\
r & \rightarrow q(X), X \in D \\
\neg r & \rightarrow \neg r \\
p(X), X \in D & \rightarrow p(X) \lor r \\
q(X) \lor \neg r & \rightarrow q(X) \lor \neg q(X), X \in D
\end{align*}
\]

Smooth FO d-DNNF
Circuit Evaluation

- Propagate weighted model count to root node
- Propagate
  - $+$ for disjunction
  - $*$ for conjunction
  - ...
  - $\sum_s \left(\binom{|D|}{s}\right) \text{wmc}(c \land |D_1^\top| = s)$
    for atom counting
- Atom counting **linear** in domain size, others **independent** of