Intelligent Agents: Web-mining Agents

Probabilistic Graphical Models

Lifted Inference

Tanya Braun
1. Recap: *Propositional* modelling
   - Factor model, Bayesian network, Markov network
   - Semantics, inference tasks + algorithms + complexity

2. *Probabilistic relational models* (PRMs)
   - Parameterised models, Markov logic networks
   - Semantics, inference tasks

3. **Lifted inference**
   - LVE, LJT, FOKC
   - Theoretical analysis

4. **Lifted learning**
   - Recap: propositional learning
   - From ground to lifted models
   - Direct lifted learning

5. **Approximate Inference: Sampling**
   - Importance sampling
   - MCMC methods

6. **Sequential models & inference**
   - Dynamic PRMs
   - Semantics, inference tasks + algorithms + complexity, learning

7. **Decision making**
   - (Dynamic) Decision PRMs
   - Semantics, inference tasks + algorithms, learning

8. **Continuous Models**
   - Probabilistic soft logic: modelling, semantics, inference tasks + algorithms
Local Symmetries and Structure

• Consider potential function as given by the table on the right

$$\phi(Travel(X), Epid, Sick(X))$$

• Only two weighted formulas \((w, \psi)\) necessary
  • \((\ln 2, \neg travel(X) \lor \neg epid \lor \neg sick(X))\)
  • \((\ln 7, travel(X) \land epid \land sick(X))\)
  • If potential of 1 instead of 2, would reduce to
    • \((\ln 7, travel(X) \land epid \land sick(X))\)
    • assignments that do not make the formula true automatically get weight of 0 = \(\ln 1\)

• If external knowledge existing, provide FOL formulas directly
  • E.g.,
    \((\ln 2, epid \land sick(X) \Rightarrow \neg travel(X))\)

Use for efficient inference

<table>
<thead>
<tr>
<th>Travel(X)</th>
<th>Epid</th>
<th>Sick(X)</th>
<th>\phi</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
<td>2</td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>true</td>
<td>2</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>false</td>
<td>2</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>true</td>
<td>2</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>false</td>
<td>2</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>true</td>
<td>2</td>
</tr>
<tr>
<td>true</td>
<td>true</td>
<td>false</td>
<td>2</td>
</tr>
<tr>
<td>true</td>
<td>true</td>
<td>true</td>
<td>7</td>
</tr>
</tbody>
</table>
MLNs: Semantics

• MLN $\Psi = \{(w_i, \psi_i)\}_{i=1}^n$, with $w_i \in \mathbb{R}$, induces a probability distribution over possible worlds $\omega \in \{\text{true, false}\}^N$

• $N = \text{the number of ground atoms in the grounded } \Psi$

$$P(\omega) = \frac{1}{Z} \prod_{i=1}^{n} \exp(w_i)^{n_i(\omega)} = \frac{1}{Z} \exp \left( \sum_{i=1}^{n} w_i n_i(\omega) \right)$$

• $n_i(\omega) = \text{number of true instances of } \psi_i \text{ in } \omega$

10 Presents(X,P,C) => Attends(X,C)

3.75 Publishes(X,C) ∧ FarAway(C) => Attends(X,C)
Outline: 3. Lifted Inference

A. Lifted variable elimination (LVE)
   - Operators
   - Algorithm
   - Complexity (including first-order dtrees), completeness, tractability
   - Variants

B. Lifted junction tree algorithm (LJT)
   - First-order junction trees (FO jtrees)
   - Algorithm
   - Complexity, completeness
   - Variants

C. First-order knowledge compilation (FOKC)
   - Normal form, circuits
   - Algorithm
   - Complexity, completeness

D. Beyond Standard Query Answering
   - Adaptive inference
   - Changing and unknown domains
   - Assignment queries
Weighted Model Counting

• Solve query answering problem by solving a weighted model counting problem
  • Weighted model count (WMC) given a sentence \( \varphi \) in propositional logic and a weight function \( \text{weight} : L \rightarrow \mathbb{R}_{\geq 0} \) associating a non-negative weight to each literal in \( \varphi \) (set \( L \)) defined by
    \[
    \text{WMC}(\varphi, \text{weight}) = \sum_{\omega \in \Omega_{\varphi}} \prod_{l \in \omega} \text{weight}(l)
    \]
  • where \( \Omega_{\varphi} \) refers to the set of worlds of \( \varphi \)
  • Probability of a world \( \omega \) of a sentence \( \varphi \) with weight function
    \[
    P(\omega) = \frac{\prod_{l \in \omega} \text{weight}(l)}{\text{WMC}(\varphi, \text{weight})} = \frac{\text{WMC}(\varphi \land \omega, \text{weight})}{\text{WMC}(\varphi, \text{weight})}
    \]
  • A query for literal \( q \) given evidence \( e \) is solved by computing
    \[
    P(q|e) = \frac{\text{WMC}(\varphi \land q \land e, \text{weight})}{\text{WMC}(\varphi \land e, \text{weight})}
    \]

Vgl. \( P(Q|E) = \frac{P(Q,E)}{P(E)} \)
Weighted Model Counting: Example

- **Sentence**
  - \( \text{sun} \land \text{rain} \Rightarrow \text{rainbow} \)

- **Weight function:**
  - \( \text{weight}(\text{sun}) = 1 \)
  - \( \text{weight}(\neg\text{sun}) = 5 \)
  - \( \text{weight}(\text{rain}) = 2 \)
  - \( \text{weight}(\neg\text{rain}) = 7 \)
  - \( \text{weight}(\text{rainbow}) = 0.1 \)
  - \( \text{weight}(\neg\text{rainbow}) = 10 \)

\[
WMC(\varphi, \text{weight}) = \sum_{\omega \in \Omega_\varphi} \prod_{l \in \omega} \text{weight}(l)
\]

<table>
<thead>
<tr>
<th>rain</th>
<th>sun</th>
<th>rainbow</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>7 \cdot 5 \cdot 10</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>7 \cdot 5 \cdot 0.1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>7 \cdot 1 \cdot 10</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>7 \cdot 1 \cdot 0.1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>2 \cdot 5 \cdot 10</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2 \cdot 5 \cdot 0.1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2 \cdot 1 \cdot 10</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2 \cdot 1 \cdot 0.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>+ 525.4</td>
</tr>
</tbody>
</table>

Each line a world \( \omega \in \Omega_\varphi \)
Weighted Model Counting: Example

• Sentence
  • \( \text{sun} \land \text{rain} \Rightarrow \text{rainbow} \)

• Weight function:
  • \( \text{weight}(\text{sun}) = 1 \)
  • \( \text{weight}(\neg \text{sun}) = 5 \)
  • \( \text{weight}(\text{rain}) = 2 \)
  • \( \text{weight}(\neg \text{rain}) = 7 \)
  • \( \text{weight}(\text{rainbow}) = 0.1 \)
  • \( \text{weight}(\neg \text{rainbow}) = 10 \)

• Probability of worlds:
  • \( P(\omega) = \frac{\prod_{l \in \omega} \text{weight}(l)}{\text{WMC}(\varphi, \text{weight})} = \frac{\text{WMC}(\varphi \land \omega, \text{weight})}{\text{WMC}(\varphi, \text{weight})} \)

  \[(\text{sun} \land \text{rain} \Rightarrow \text{rainbow}) \land \text{sun} \land \text{rain} \land \text{rainbow}\]

<table>
<thead>
<tr>
<th>rain</th>
<th>sun</th>
<th>rainbow</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>7 \cdot 5 \cdot 10 = 350</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>7 \cdot 5 \cdot 0.1 = 3.5</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>7 \cdot 1 \cdot 10 = 70</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>7 \cdot 1 \cdot 0.1 = 0.7</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>2 \cdot 5 \cdot 10 = 100</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2 \cdot 5 \cdot 0.1 = 1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2 \cdot 1 \cdot 10 = 200</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2 \cdot 1 \cdot 0.1 = 0.2</td>
</tr>
</tbody>
</table>

\( \omega = (\text{sun}, \text{rain}, \text{rainbow}) \in \Omega_\varphi \)
Weighted Model Counting: Example

• Sentence
  • \(\text{sun} \land \text{rain} \Rightarrow \text{rainbow}\)

• Weight function:
  • \(\text{weight}(\text{sun}) = 1\)
  • \(\text{weight}(\neg \text{sun}) = 5\)
  • \(\text{weight}(\text{rain}) = 2\)
  • \(\text{weight}(\neg \text{rain}) = 7\)
  • \(\text{weight}(\text{rainbow}) = 0.1\)
  • \(\text{weight}(\neg \text{rainbow}) = 10\)

• Probability of worlds:
  • \(P(\text{rain}) = \frac{\text{WMC}(\varphi \land \text{rain}, \text{weight})}{\text{WMC}(\varphi, \text{weight})}\)

\[
P(q) = \frac{\text{WMC}(\varphi \land q, \text{weight})}{\text{WMC}(\varphi, \text{weight})}
\]

\(\text{(sun} \land \text{rain} \Rightarrow \text{rainbow}) \land \text{rain}\)

<table>
<thead>
<tr>
<th>rain</th>
<th>sun</th>
<th>rainbow</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>7 \cdot 5 \cdot 10</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>7 \cdot 5 \cdot 0.1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>7 \cdot 1 \cdot 10</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>7 \cdot 1 \cdot 0.1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>2 \cdot 5 \cdot 10</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2 \cdot 5 \cdot 0.1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2 \cdot 1 \cdot 10</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2 \cdot 1 \cdot 0.1</td>
</tr>
</tbody>
</table>

\(\text{All } \omega \in \Omega_\varphi \text{ where } \text{rain} \text{ holds}\)
WMC and Inference

• Solving a WMC problem for a sentence $\varphi$ as introduced on previous slides is exponential in number of worlds with probability $> 0$ (models)

• To be more efficient, build a helper structure
  • Bring sentence into negation normal form (NNF)
    • NNF: Formulas contain only negations directly in front of variables, conjunctions, and disjunctions
  • E.g.,
    • $\text{sun} \land \text{rain} \Rightarrow \text{rainbow}$
      \[ \equiv \neg (\text{sun} \land \text{rain}) \lor \text{rainbow} \quad \text{(Apply De Morgan’s law)} \]
      \[ \equiv \neg \text{sun} \lor \neg \text{rain} \lor \text{rainbow} \quad \text{(NNF)} \]
Circuits

• Represent the NNF sentence as a directed, acyclic graph called circuit with leaves labelled with literals ($l$ or $\neg l$) or $true$, $false$ with inner nodes being
  • **Deterministic** disjunctions
    • Only one disjunct (child node) can be true at the same time
      • I.e., their conjunction is unsatisfiable
  • **Decomposable** conjunctions
    • Each pair of conjuncts (child nodes) must be independent
      • I.e., they cannot share any variables

• Circuit is then in **d-DNNF**
  • deterministic Decomposable NNF
  • See later why important
Circuits: Example

- **Deterministic disjunctions**
  - Only one disjunct (child node) can be true at the same time
    - I.e., their conjunction is unsatisfiable

- **Decomposable conjunctions**
  - Each pair of conjuncts (child nodes) must be independent
    - I.e., they cannot share any variables

- E.g., \( \neg \text{sun} \lor \neg \text{rain} \lor \text{rainbow} \)
  - \(<\text{disjunct}> \lor \text{rainbow}\)
    - Determinism:
      \(<\text{disjunct}>\) can only be true if \(\neg \text{rainbow}\) is not
      - Add \(\neg \text{rainbow}\) to disjunct:
        \(\neg \text{rainbow} \land <\text{disjunct}>\)
Circuits: Example

- Deterministic disjunctions
  - Only one disjunct (child node) can be true at the same time
    - I.e., their conjunction is unsatisfiable
- Decomposable conjunctions
  - Each pair of conjuncts (child nodes) must be independent
    - I.e., they cannot share any variables
- E.g., $\neg$sun $\lor$ $\neg$rain $\lor$ rainbow
  - $<$disjunct$>$ $\lor$ rainbow
    - Determinism: $<$disjunct$>$ can only be true if rainbow is not
      - Add $\neg$rainbow to disjunct: $\neg$rainbow $\land$ $<$disjunct$>$
      - $<$disjunct$>$ now part of a conjunction with $\neg$rainbow
        - Decomposability: May not contain Rainbow
Circuits: Example

- Deterministic disjunctions
  - Only one disjunct (child node) can be true at the same time
    - I.e., their conjunction is unsatisfiable
- Decomposable conjunctions
  - Each pair of conjuncts (child nodes) must be independent
    - I.e., they cannot share any variables

- E.g., \( \neg \text{sun} \lor \neg \text{rain} \lor \text{rainbow} \)
- \(<\text{disjunct}> \lor \neg \text{rain}\)
  - Determinism:
    - \(<\text{disjunct}>\) can only be true if \(\neg \text{rain}\) is not, i.e., if \(\text{rain}\) is
    - Add \(\text{rain}\) to disjunct:
      - \(\text{rain} \land <\text{disjunct}>\)
Circuits: Example

• Deterministic disjunctions
  • Only one disjunct (child node) can be true at the same time
    • I.e., their conjunction is unsatisfiable

• Decomposable conjunctions
  • Each pair of conjuncts (child nodes) must be independent
    • I.e., they cannot share any variables

• E.g., \( \neg \text{sun} \lor \neg \text{rain} \lor \text{rainbow} \)
  • <disjunct> \( \lor \neg \text{rain} \)
    • Determinism:
      <disjunct> can only be true if
      \( \neg \text{rain} \) is not, i.e., if \text{rain} is
      • Add \text{rain} to disjunct:
        \text{rain} \land <\text{disjunct}>
      • <disjunct> now part of a conjunction with \text{rain}
        • Decomposability: May not contain \text{Rain} <\text{disjunct}>
Circuits: Example

- Deterministic disjunctions
  - Only one disjunct (child node) can be true at the same time
    - I.e., their conjunction is unsatisfiable

- Decomposable conjunctions
  - Each pair of conjuncts (child nodes) must be independent
    - I.e., they cannot share any variables

- E.g., \( \neg \text{sun} \lor \neg \text{rain} \lor \text{rainbow} \)

- Add as conjunct
  - Decomposability: Does not share variables with sibling node
Effects of d-DNNF

• Effects of d-DNNF
  • Deterministic disjunctions
    • Only one disjunct (child node) can be true at the same time
      • I.e., their conjunction is unsatisfiable
  • Assume children $c_i, c_j$ represent probabilities $p_i, p_j$
    • Node then represents probability of $P(c_i \lor c_j)$
      • $P(c_i \lor c_j) = P(c_i) + P(c_j) - P(c_i \land c_j)$
      • If only $c_i$ or $c_j$ can be true at a time, $P(c_i \land c_j) = 0$, i.e.,
        • $P(c_i \lor c_j) = P(c_i) + P(c_j)$
    • Can replace $\lor$ with $+$ for inference calculations
Effects of d-DNNF

• Effects of d-DNNF
  • Decomposable conjunctions
    • Each pair of conjuncts (child nodes) must be independent
      • I.e., they cannot share any variables
  • Assume children $c_i, c_j$ represent probabilities $p_i, p_j$
    • Node then represents probability of $P(c_i \land c_j)$
    • If $c_i$ and $c_j$ independent (decomposable), then $P(c_i \land c_j) = P(c_i) \cdot P(c_j)$
  • Can replace $\land$ with $\cdot$ for inference calculations
Smooth d-DNNF (sd-DNNF)

- Smooth circuits: constant runtime for certain queries
  - Any pair of disjuncts mentions the same set of variables
  - E.g., $\neg\text{sun} \lor \neg\text{rain} \lor \text{rainbow}$
    - Two disjunctions that do not fulﬁl the smoothness property

- Rules for conversion
  - For each negation of a positive literal $l$ not appearing, replace $l$ by $l \lor (\neg l \land \text{false})$
  - For each variable $A$ not mentioned in a disjunct $\langle\text{disjunct}\rangle$, add $a \lor \neg a$ with a conjunction to $\langle\text{disjunct}\rangle$: $\langle\text{disjunct}\rangle \land (a \lor \neg a)$
Smooth d-DNNF (sd-DNNF)

- Add $sun \lor \neg sun$ to $\neg rain$, replacing $\neg rain$ with

$$\neg rain \land (sun \lor \neg sun)$$
Smooth d-DNNF (sd-DNNF)

- Add \( \text{sun} \lor \neg \text{sun} \) and \( \text{rain} \lor \neg \text{rain} \), replacing \( \text{rainbow} \) with

\[
\text{rainbow} \land (\text{sun} \lor \neg \text{sun}) \land (\text{rain} \lor \neg \text{rain})
\]
Circuit for Model Counting

• Model counting problem: Count how many models fulfil a sentence

• Model counting arithmetic circuit
  • Replace $\&$ with $\cdot$
  • Replace $\lor$ with $+$
  • Replace leaves with 1’s
Circuit for Model Counting

• Propagate 1’s upwards (from leaves to root), using arithmetic operations in inner nodes to combine incoming numbers
  • Result at root: Model count

<table>
<thead>
<tr>
<th></th>
<th>rain</th>
<th>sun</th>
<th>rainbow</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Conditioning

- To get model count of models fulfilling certain truth values
  - Replace 1’s with zeros where literal contradicts truth values
    - Could minimise circuit
  - E.g., condition on \(-rainbow\)
Circuit for Weighted Model Counting

- Replace literals with weights in leaves and propagate weights upwards
  - Computes $WMC(\varphi, \text{weight})$

Weight assignments:
- $\text{weight}(\text{sun}) = 1$
- $\text{weight}(\neg\text{sun}) = 5$
- $\text{weight}(\text{rain}) = 2$
- $\text{weight}(\neg\text{rain}) = 7$
- $\text{weight}(\text{rainbow}) = 0.1$
- $\text{weight}(\neg\text{rainbow}) = 10$
Circuit for Weighted Model Counting

- For probabilities of worlds or query terms $\omega$, condition on truth values
  1. Compute $WMC(\varphi, weight)$
  2. Compute $WMC(\varphi \land \omega, weight)$
  3. Divide the two counts

$P(\omega = \{sun, rain, rainbow\})$

\[
\frac{WMC(\varphi \land \omega, weight)}{WMC(\varphi, weight)} = \frac{0.2}{525.4} = 0.00038
\]
Knowledge Compilation

- Solve the weighted model counting problem by knowledge compilation

- Given a theory $\Delta$ and a set of queries $\{P(q_i|e)\}_{i=1}^{m}$
  - Build a circuit for theory $\Delta$ (a conjunction of sentences)
  - Make the circuit a WMC circuit
    - Replace inner nodes with arithmetic operations
    - Replace leaves with weights
  - Condition on given evidence $e$
    - Replace weights with 0 where literals contradict $e$
  - Calculate $WMC(\Delta \land e, \text{weight})$ in the circuit
    - By propagating the weights upwards
  - For each query $P(q_i|e)$ in the circuit
    - Compute $WMC(\Delta \land e \land q_i, \text{weight})$
    - Return or store $P(q_i|e) = \frac{WMC(\Delta \land e \land q_i, \text{weight})}{WMC(\Delta \land e, \text{weight})}$
Propositional ➔ First-order

• If input theory is in FOL-DC ((function-free) first-order logic with domain constraints), one could ground the theory given domains and build a circuit for the grounded theory
  • FOL-DS includes intensional conjunctions and disjunctions ($\forall$, $\exists$)
  • Leads to repeated structures in circuit
• Combine repeated structures using new inner node types for intensional conjunctions and disjunctions ($\forall$, $\exists$)
• We are not going into every detail of FOKC;
  • For complete description, analysis, and discussion, see the PhD thesis by Guy Van den Broeck
Weighted First-order Model Counting

- Define a weighted first-order model counting problem using a weighted first-order model count (WFOMC)

\[
WFOMC(\Delta, w_T, w_F) = \sum_{\omega = \omega_T \cup \omega_F} \prod_{l \in \omega_T} w_T(\text{pred}(l)) \prod_{l \in \omega_F} w_F(\text{pred}(l))
\]

- \(\Delta\) a theory in FOL-DC
- \(w_T\) a weight function for predicates being positive
- \(w_F\) a weight function for predicates being negative
- \(\Omega_\Delta\) the set of worlds (i.e., models in logics) of \(\Delta\)
- \(\text{pred}(l)\) a function mapping a literal \(l\) to its predicate

- Query can be answered by computing

\[
P(q_i | e) = \frac{WFOMC(\Delta \land e \land q_i, w_T, w_F)}{WFOMC(\Delta \land e, w_T, w_F)}
\]
• Theory: one sentence
  \( \forall X \in \text{People} : \) \( \text{smokes}(X) \Rightarrow \text{cancer}(X) \)
  
  • People = \{x_1, x_2\}

• Weight functions
  • \( w_T(\text{smokes}(X)) = 3 \)
  • \( w_F(\neg \text{smokes}(X)) = 1 \)
  • \( w_T(\text{cancer}(X)) = 6 \)
  • \( w_F(\neg \text{cancer}(X)) = 2 \)

• Model count: 9
  • Worlds that fulfil the theory

\[
WFOMC(\Delta, w_T, w_F) = \sum_{\omega = \omega_T \cup \omega_F} \prod_{l \in \omega_T} w_T(\text{pred}(l)) \prod_{l \in \omega_F} w_F(\text{pred}(l))
\]

<table>
<thead>
<tr>
<th>(x_1)</th>
<th>(c(x_1))</th>
<th>(x_2)</th>
<th>(c(x_2))</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1 \cdot 2 \cdot 1 \cdot 2</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1 \cdot 2 \cdot 1 \cdot 6</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1 \cdot 2 \cdot 3 \cdot 2</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1 \cdot 6 \cdot 1 \cdot 2</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1 \cdot 6 \cdot 1 \cdot 6</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1 \cdot 6 \cdot 3 \cdot 6</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3 \cdot 2 \cdot 1 \cdot 2</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>3 \cdot 2 \cdot 1 \cdot 6</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>3 \cdot 2 \cdot 3 \cdot 2</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>3 \cdot 2 \cdot 3 \cdot 6</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>3 \cdot 6 \cdot 1 \cdot 2</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>3 \cdot 6 \cdot 1 \cdot 6</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>3 \cdot 6 \cdot 3 \cdot 2</td>
</tr>
</tbody>
</table>
| 1      | 1      | 1      | 1      | 3 \cdot 6 \cdot 3 \cdot 6 | 324   | +  676
• Theory: one sentence
  \( \forall X \in \text{People} : \) \( \text{smokes}(X) \Rightarrow \text{cancer}(X) \)
• People = \( \{x_1, x_2\} \)
• Weight functions
  \( w_T(\text{smokes}(X)) = 3 \)
  \( w_F(\neg \text{smokes}(X)) = 1 \)
  \( w_T(\text{cancer}(X)) = 6 \)
  \( w_F(\neg \text{cancer}(X)) = 2 \)

\[
P(s(x_1)) = \frac{\text{WFOMC}(\Delta \land s(x_1), w_T, w_F)}{\text{WFOMC}(\Delta, w_T, w_F)} = \frac{36 + 108 + 324}{676} = 0.692
\]
First-order (FO) Circuits

• Assume theory in Skolem normal form + CNF
  • Sequence of intensional conjunctions in CNF
  • E.g., with \( s = \text{smokes}, c = \text{cancer} \)
    \[
    \forall X \in \text{People} : s(X) \Rightarrow c(X) \\
    \equiv \forall X \in \text{People} : \neg s(X) \lor c(X)
    \]

• FO circuit (extract)
  • Inner nodes:
    • Extensional conjunctions/disjunctions (as before)
    • Set conjunctions
  • Leaf nodes
    • Positive and negative predicates, \( \text{true, false} \)
  • Full + construction:
    see PhD thesis by Guy Van den Broeck
Smooth FO d-DNNF Circuits

• Properties
  • Deterministic disjunctions
    • Only one disjunct (child node) can be true at the same time
  • Decomposable conjunctions
    • Each pair of conjuncts (child nodes) must be independent
  • Smoothness
    • Each disjunct contains the same variables

\[ \forall X \left( X \in \text{People} \right) \]
\[ \lor \]
\[ c(X) \]
\[ \land \]
\[ \neg s(X), \neg c(X) \]

\[ \forall X \left( X \in \text{People} \right) \]
\[ \lor \]
\[ c(X), s(X), \neg s(X), \neg c(X) \]
Arithmetic FO d-DNNF Circuits

- Replace
  - Replace $\land$ with $\cdot$
  - Replace $\lor$ with $+$
  - Replace $\forall$ with exponentiation for $|\text{Domain}|$
  - Replace leaves with 1’s
- E.g., with $|\text{People}| = |\{x_1, x_2\}| = 2$
WFOMC Circuits

- Replace
  - Replace $\land$ with $\cdot$
  - Replace $\lor$ with $+$
  - Replace $\forall$ with exponentiation for $|\text{Domain}|$
  - Replace leaves with weights
- E.g., with $|\text{People}| = |\{x_1, x_2\}| = 2$

$$WFOMC(\Delta, w_T, w_F) = \sum_{\omega=\omega_T \cup \omega_F} \prod_{l \in \omega_T} w_T(\text{pred}(l)) \prod_{l \in \omega_F} w_F(\text{pred}(l))$$

| People | $w_T(\text{smokes}(X)) = 3$
|--------|-------------------
|        | $w_F(\neg \text{smokes}(X)) = 1$
|        | $w_T(\text{cancer}(X)) = 6$
|        | $w_F(\neg \text{cancer}(X)) = 2$

Diagram:
- $\forall X \in \text{People}$
- $|\text{People}|$
- $\land$
- $\lor$
- $\neg$
- $\neg$
- $c(X)$
- $s(X)$
- $\neg s(X)$
- $\neg c(X)$
WFOMC Circuits

• Given $P(q_i|e)$
  • Basically, compile a circuit for $\Delta \land e \land q_i$ reusing components from the circuit of $\Delta \land e$
  • E.g., $P(s(x_1))$ with $|\text{People}| = |\{x_1, x_2\}| = 2$

\[
P(s(x_1)) = \frac{\text{WFOMC}(\Delta \land s(x_1), w_T, w_F)}{\text{WFOMC}(\Delta, w_T, w_F)} = \frac{468}{676} = 0.692
\]
Conditioning in FO Circuits

• Evidence on
  • Propositional variables $L$
    • Replace leaf values with 0 where literal contradicts observation
    • As in propositional circuits
  • Unary variable $L(X)$
    • For each variable $L(X)$ that one wants to condition on,
      • Replace FOL-DC formula with three copies with additional domain constraints, possibly simplify formula based on observation
        1. $X \in D_T$ for observations $l(x)$
        2. $X \in D_{\perp}$ for observations $\neg l(x)$
        3. $X \notin D_T \land X \notin D_{\perp}$ no observations
  • Compile a circuit for the extended theory
  • Given specific evidence, domains for $D_T, D_{\perp}$ are determined
    • Might be empty
  • Binary variable $L(X, Y)$
    • Can compile a circuit, no longer polynomial in time (reduction of #2SAT problem)
Conditioning in FO Circuits

• E.g., $\forall X \in \text{People} : s(X) \Rightarrow c(X)$ and $S(X)$
  1. $\forall X \in \text{People}_\top : s(X) \Rightarrow c(X)$ if $s(X)$
     $\equiv \forall X \in \text{People}_\top : c(X)$
  2. $\forall X \in \text{People}_\bot : s(X) \Rightarrow c(X)$ if $s(X)$
     $\equiv \forall X \in \text{People}_\bot : \text{true}$
  3. $\forall X \in \text{People}, X \notin \text{People}_\top, X \notin \text{People}_\bot : s(X) \Rightarrow c(X)$

• Delete Formula 2 as it is always true

• If one also wants to condition on $C(X)$, theory becomes larger again:
  • Formulas (1) and (3) contain $C(X)$ and therefore need to be replaced by three formulas, then simplify

\[
\begin{array}{c}
\forall X \\
X \in \text{People}_\top
\end{array}
\quad
\begin{array}{c}
\forall X \\
X \in \text{People}'
\end{array}
\quad
\begin{array}{c}
\lor \\
\land
\end{array}
\quad
\begin{array}{c}
c(X)
\end{array}
\quad
\begin{array}{c}
c(X)
\end{array}
\quad
\begin{array}{c}
s(X)
\end{array}
\quad
\begin{array}{c}
\neg s(X)
\end{array}
\quad
\begin{array}{c}
\neg c(X)
\end{array}
\]
**First-order Knowledge Compilation (FOKC)**

- **Solve** the weighted first-order model counting problem by knowledge compilation

**Given**
- a theory $\Delta$ in FOL-DC in Skolem NNF
- a weight function $w_T$ for predicates being positive
- a weight function $w_F$ for predicates being negative
- and a set of queries $\{P(q_i|e)\}_{i=1}^{m}$ with evidence for variables $E$

**Do**
- Build a WFOMC circuit $C_\Delta$ for $\Delta$, also preparing for evidence on $E$
- Condition on $e$
- Calculate $WFOMC(\Delta \land e, w_T, w_F)$ in $C_\Delta$
- For each query $P(q_i|e)$
  - Build a WFOMC circuit $C_{\Delta,q_i}$ for $\Delta \land q_i$ conditioned on $e$
  - Compute $WFOMC(\Delta \land e \land q_i, w_T, w_F)$ in $C_{\Delta,q_i}$
  - Return or store $P(q_i|e) = \frac{WFOMC(\Delta \land e \land q_i, w_T, w_F)}{WFOMC(\Delta \land e, w_T, w_F)}$
MLNs for WFOMCs

• Weights in MLNs specified for formulas instead of single predicates
  • E.g., example from the beginning
    • \((\ln 7 , \text{travel}(X) \land \text{epid} \land \text{sick}(X))\)
    • \((\ln 2 , \neg\text{travel}(X) \lor \neg\text{epid} \lor \neg\text{sick}(X))\)

• Trick:
  • Introduce a new predicate \(\theta_i\) containing all free variables of \(\psi_i\) as equivalent to \(\psi_i\)
  • E.g.,
    • \(\forall X \in \text{People} : \theta_1(X) \iff (\text{travel}(X) \land \text{epid} \land \text{sick}(X))\)
    • \(\forall X \in \text{People} : \theta_2(X) \iff (\neg\text{travel}(X) \lor \neg\text{epid} \lor \neg\text{sick}(X))\)
  • Specify weight functions such that \(\theta_i\) takes the weight of \(\psi_i\)
    • \(w_T(\theta_1(X)) = \exp(\ln 7) = 7\)
    • \(w_T(\theta_2(X)) = \exp(\ln 2) = 2\)
    • All other predicates and \(\neg\theta_1, \neg\theta_2\) are mapped to 1 by both \(w_T, w_F\)
WFOMC Reduction

• Formally, given an MLN $\Psi = \{(w_i, \psi_i)\}_{i=1}^{n}$
  • Transform each weighted formula $(w_i, \psi_i)$ into an FOL-DC formula
    $$\forall X_i, cs_i : \theta_i(X_i) \Leftrightarrow \psi_i$$
  • where
    • $X_i$ are the free variables in $\psi_i$
    • $cs_i$ is the constraint set that enforces the domain constraints as given by the MLN
    • $\theta_i(X_i)$ is a new predicate containing all free variables of $\psi_i$
  • Specify weight functions $w_T, w_F$ such that for each
    • $w_T(\theta_i(X_i)) = \exp(w_i)$
    • $w_T(p_i) = 1$ for all predicates $p_i$ occurring in $\Psi$
    • $w_F(\theta_i(X_i)) = w_F(p_i) = 1$
  • Continue with knowledge compilation
Example

• Given
  • $\ln(7, \text{travel}(X) \land \text{epid} \land \text{sick}(X))$
  • $\ln(2, \neg\text{travel}(X) \lor \neg\text{epid} \lor \neg\text{sick}(X))$

• Resulting theory
  • with $t = \text{travel}, e = \text{epid}, s = \text{sick}$
    • $\forall X \in \text{People} : \theta_1(X) \iff (t(X) \land e \land s(X))$
    • $\forall X \in \text{People} : \theta_2(X) \iff (\neg t(X) \lor \neg e \lor \neg s(X))$
  • with weight functions
    • $w_T(\theta_1(X)) = 7$
    • $w_T(\theta_2(X)) = 2$
    • Rest mapped to 1 by both $w_T, w_F$

• Transform formulas into CNF
Example: Normal Form

• Transform formulas into CNF
  
  • \( \forall X \in \text{People} : \theta_1(X) \equiv (t(X) \land e \land s(X)) \)
    
    \( \theta_1(X) \equiv (t(X) \land e \land s(X)) \)  
    (resolve \( \equiv \))
    
    \( \equiv (\theta_1(X) \Rightarrow (t(X) \land e \land s(X))) \land (\theta_1(X) \Leftarrow (t(X) \land e \land s(X))) \)  
    (De Morgan on \( \Rightarrow \))
    
    \( \equiv (\neg \theta_1(X) \lor (t(X) \land e \land s(X))) \land (\theta_1(X) \lor \neg (t(X) \land e \land s(X))) \)  
    (move \( \neg \) inward)
    
    \( \equiv (\neg \theta_1(X) \lor (t(X) \land e \land s(X))) \land (\theta_1(X) \lor \neg t(X) \lor \neg e \lor \neg s(X)) \)  
    (distribute \( \lor \))
    
    \( \equiv (\neg \theta_1(X) \lor t(X)) \land (\neg \theta_1(X) \lor e) \land (\neg \theta_1(X) \lor s(X)) \land (\theta_1(X) \lor \neg t(X) \lor \neg e \lor \neg s(X)) \)  
    (CNF)

• Result (each conjunct as own formula):
  
  • \( \forall X \in \text{People} : \neg \theta_1(X) \lor t(X) \)
  
  • \( \forall X \in \text{People} : \neg \theta_1(X) \lor e \)
  
  • \( \forall X \in \text{People} : \neg \theta_1(X) \lor s(X) \)
  
  • \( \forall X \in \text{People} : \theta_1(X) \lor \neg t(X) \lor \neg e \lor \neg s(X) \)
Example: Normal Form

• Transform formulas into CNF

  • \( \forall X \in \text{People} : \theta_2(X) \iff (\neg t(X) \lor \neg e \lor \neg s(X)) \)
    \[ \theta_2(X) \iff (\neg t(X) \lor \neg e \lor \neg s(X)) \equiv (\neg \theta_2(X) \lor \neg t(X) \lor \neg e \lor \neg s(X)) \land (\theta_2(X) \lor (\neg t(X) \lor \neg e \lor \neg s(X))) \]
    \[ \equiv (\neg \theta_2(X) \lor \neg t(X) \lor \neg e \lor \neg s(X)) \land (\theta_2(X) \lor (t(X) \land e \land s(X))) \]
    \[ \equiv (\neg \theta_2(X) \lor \neg t(X) \lor \neg e \lor \neg s(X)) \land (\theta_2(X) \lor t(X)) \land (\theta_2(X) \lor e) \land (\theta_2(X) \lor s(X)) \]

• Result (each conjunct as own formula):
  • \( \forall X \in \text{People} : \neg \theta_2(X) \lor \neg t(X) \lor \neg e \lor \neg s(X) \)
  • \( \forall X \in \text{People} : \theta_2(X) \lor t(X) \)
  • \( \forall X \in \text{People} : \theta_2(X) \lor e \)
  • \( \forall X \in \text{People} : \theta_2(X) \lor s(X) \)
Example: FO d-DNNF Circuit

- Given theory in CNF
  - $\forall X \in \text{People} : \neg \theta_1(X) \lor t(X)$
  - $\forall X \in \text{People} : \neg \theta_1(X) \lor e$
  - $\forall X \in \text{People} : \neg \theta_1(X) \lor s(X)$
  - $\forall X \in \text{People} : \theta_1(X) \lor \neg t(X) \lor \neg e \lor \neg s(X)$
  - $\forall X \in \text{People} : \neg \theta_2(X) \lor \neg t(X) \lor \neg e \lor \neg s(X)$
  - $\forall X \in \text{People} : \theta_2(X) \lor t(X)$
  - $\forall X \in \text{People} : \theta_2(X) \lor e$
  - $\forall X \in \text{People} : \theta_2(X) \lor s(X)$

- Resulting FO d-DNNF circuit generated by the FOKC implementation
  - Some leaves repeated for readability
Example: FO d-DNNF Circuit

• Given theory in CNF
  1. \( \forall X \in \text{People} : \neg \theta_2(X) \lor \neg t(X) \lor \neg s(X) \lor \neg e \)
  2. \( \forall X \in \text{People} : \theta_1(X) \lor \neg t(X) \lor \neg e \lor \neg s(X) \)
  3. \( \forall X \in \text{People} : \neg \theta_1(X) \lor t(X) \)
  4. \( \forall X \in \text{People} : \neg \theta_1(X) \lor e \)
  5. \( \forall X \in \text{People} : \neg \theta_1(X) \lor s(X) \)
  6. \( \forall X \in \text{People} : \theta_2(X) \lor t(X) \)
  7. \( \forall X \in \text{People} : \theta_2(X) \lor e \)
  8. \( \forall X \in \text{People} : \theta_2(X) \lor s(X) \)
Example: FO d-DNNF Circuit

• Given theory in CNF

1. \( \forall X \in \text{People} : \neg \theta_2(X) \lor \neg t(X) \lor \neg s(X) \lor \neg e \)
2. \( \forall X \in \text{People} : \theta_1(X) \lor \neg t(X) \lor \neg e \lor \neg s(X) \)
3. \( \forall X \in \text{People} : \neg \theta_1(X) \lor t(X) \)
4. \( \forall X \in \text{People} : \neg \theta_1(X) \lor e \)
5. \( \forall X \in \text{People} : \neg \theta_1(X) \lor s(X) \)
6. \( \forall X \in \text{People} : \theta_2(X) \lor t(X) \)
7. \( \forall X \in \text{People} : \theta_2(X) \lor e \)
8. \( \forall X \in \text{People} : \theta_2(X) \lor s(X) \)
Example: FO d-DNNF Circuit

• Given theory in CNF
  1. \( \forall X \in \text{People} : \neg \theta_2(X) \lor \neg t(X) \lor \neg s(X) \lor \neg e \)
  2. \( \forall X \in \text{People} : \theta_1(X) \lor \neg t(X) \lor \neg e \lor \neg s(X) \)
  3. \( \forall X \in \text{People} : \neg \theta_1(X) \lor t(X) \)
  4. \( \forall X \in \text{People} : \neg \theta_1(X) \lor e \)
  5. \( \forall X \in \text{People} : \neg \theta_1(X) \lor s(X) \)
  6. \( \forall X \in \text{People} : \theta_2(X) \lor t(X) \)
  7. \( \forall X \in \text{People} : \theta_2(X) \lor e \)
  8. \( \forall X \in \text{People} : \theta_2(X) \lor s(X) \)

• Not smooth since
  • Right branch of root \( \lor \) misses \( s(X), t(X) \)
  • Right branch of \( \lor \) after set conjunction misses \( t(X) \)
Example: Smoothed FO d-DNNF Circuit

As generated by the FOKC implementation
Theoretical Results

• Compilation independent of domain sizes
  • Just like construction of FO jtree is also independent of domain sizes

• Inference
  • Polynomial in domain sizes
    • Based on the computations that are computed at different node types

• Completeness as before
  • $\mathcal{M}^{2lv}$
    • Two-logvar theories with max. two logical variables per formula
  • $\mathcal{M}^{1prv}$
    • One logvar per variable
Implementation

• Available at
  • [https://github.com/UCLA-StarAI/Forclift](https://github.com/UCLA-StarAI/Forclift)
    • May no longer work according to Guy so you may have to try
      • [https://github.com/tanyabraun/wfomc](https://github.com/tanyabraun/wfomc)
  • Officially three input formats
    • Based on the normal form required (.wmc)
    • Early version of parfactor graphs (.fg)
    • MLN version (.mln)
    → MLN file format only one I got the compiled version to parse
Implementation

• Query answering times, trade-off criteria

• Increasing domain size

- FOKC almost invariant w.r.t. domain sizes

• Increasing counting width

- FOKC does not build histograms, which blow up the representation

Runtimes in milliseconds
Probabilistic Theorem Proving (PTP)

• Based on theorem proving in logics
• Solves lifted weighted model counting problem
  • Similar to the weighted first-order model counting problem by Guy Van den Broeck
  • MLNs as input

• Implementation available: Alchemy
  • http://alchemy.cs.washington.edu
  • Input format: MLNs
LJT as a Framework

• Remember: LJT only specifies a helper structure and steps
  • I.e., no specific inference algorithm as a subroutine for its calculations

• Requirements for subroutine
  • Lifted evidence handling
  • Lifted message calculation
    • Message = conj. param’d query
  • Lifted query answering

• LJTKC: LJT with LVE & FOKC
  • LVE for evidence entering and message passing
  • FOKC for query answering
    • Only for Boolean PRVs

<table>
<thead>
<tr>
<th>Calculated Lifted?</th>
<th>LVE</th>
<th>FOKC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Evidence</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Messages</td>
<td>✓</td>
<td>✗*</td>
</tr>
<tr>
<td>Queries</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

* Not obvious how parameterised queries are handled in circuits

LJTKC: Algorithm

**LJTKC** \( (G, \{Q_i\}_{i=1}^n, \{g_e\}_{e=1}^m) \)

- Construct an FO jtree \( J \) for \( G \)
- Enter evidence \( \{g_e\}_{e=1}^m \) into \( J \)
- Pass message in \( J \)

**for each parcluster** \( C_j \) **in** \( J \) **do**

- Transform local model \( G_j \) into an MLN \( \Psi_j \)
- Transform \( \Psi_j \) into a theory \( \Delta_j \) in CNF with
  - weight functions \( w_T, w_F \)
- Build a circuit \( C_j \) for \( \Delta_j \)
- Compute \( c_j = WFOMC(\Delta_j, w_T, w_F) \) in \( C_j \)

**for each query terms** \( Q_i \) **do**

- Build a circuit \( C_{j,q} \) for \( \Delta_j \land q_i \)
- Compute \( c_q = WFOMC(\Delta_j \land q_i, w_T, w_F) \) in \( C_{j,q} \)
- Return or store \( \frac{c_q}{c_j} \) (and possibly \( 1 - \frac{c_q}{c_j} \))
Summary

• Propositional (weighted) model counting
  • WMC definition
  • Circuits:
    • Inner nodes: conjunctions/disjunctions
    • Leaves: literals, true, false
    • Properties: d-DNNF, smooth
    • Model counts, WMC by propagation
  • Knowledge compilation
    • Inference in circuits:
      Query answering by weighted model counting in circuits

• Lifted (weighted) model counting
  • WFOMC definition
  • FO circuits
    • Inner nodes can also be set conjunctions/disjunctions
  • First-order knowledge compilation
    • Inference in FO circuits

• Further uses
  • WFOMC in PTP
  • FOKC for query answering in LJT
Outline: 3. Lifted Inference

A. Lifted variable elimination (LVE)
   • Operators
   • Algorithm
   • Complexity (including first-order dtrees), completeness, tractability
   • Variants

B. Lifted junction tree algorithm (LJT)
   • First-order junction trees (FO jtrees)
   • Algorithm
   • Complexity, completeness
   • Variants

C. First-order knowledge compilation (FOKC)
   • Normal form, circuits
   • Algorithm
   • Complexity, completeness

D. Beyond Standard Query Answering
   • Adaptive inference
   • Changing and unknown domains
   • Assignment queries