Probabilistic Theorem Proving

Vibhav Gogate

(Joint work with Pedro Domingos)
Progress to Date

- First-order variable elimination
  [Poole, 2003; de Salvo Braz, 2007; etc.]

- Lifted belief propagation
  [Singla & Domingos, 2008; etc.]

- Hot topic in UAI, IJCAI, etc.
Limitations

- FOVE does not scale
- Lifted BP cannot handle determinism, etc.
- Neither exploits logical structure
- Unclear relation to logical inference
Probabilistic Theorem Proving addresses all of these limitations.

Probabilistic Theorem Proving is a single inference procedure with graphical model inference and theorem proving as special cases.
Probabilistic Theorem Proving

**Given** Probabilistic knowledge base $K$
Query formula $Q$

**Output** $P(Q|K)$
Propositional Logic

- **Atoms**: Symbols representing propositions
- **Logical connectives**: ¬, ∧, ∨, etc.
- **Knowledge base**: Set of formulas
- Every KB can be converted to **CNF**
  - **CNF**: Conjunction of clauses
  - **Clause**: Disjunction of literals
  - **Literal**: Atom or its negation
- **Entailment**: Does KB entail query?
Theorem Proving

\[ TP(KB, \text{Query}) \]

\[ KB_Q \leftarrow KB \cup \{ \neg \text{Query} \} \]

\[ \text{return } \neg\text{SAT(CNF}(KB_Q)) \]
Satisfiability (DPLL)

\[
\text{SAT}(CNF) \\
\begin{array}{l}
\text{if } CNF \text{ is empty return True} \\
\text{if } CNF \text{ contains empty clause return False} \\
\text{choose an atom } A \\
\text{return SAT}(CNF(A)) \lor SAT(CNF(\neg A))
\end{array}
\]
First-Order Logic

- Atom: Predicate(Variables, Constants)
  E.g.: $\text{Friends}(\text{Anna}, x)$

- Ground atom: All arguments are constants

- Quantifiers: $\forall, \exists$

- This talk: Finite, Herbrand interpretations
First-Order Theorem Proving

- **Propositionalization**
  1. Form all possible ground atoms
  2. Apply propositional theorem prover

- **Lifted Inference: Resolution**
  - Resolve pairs of clauses until empty clause derived
  - Unify literals by substitution, e.g.: $x = Bob$ unifies $Friends(Anna, x)$ and $Friends(Anna, Bob)$

\[
\neg Friends(Anna, x) \lor Happy(x) \\
\hline
Friends(Anna, Bob) \\
\hline
Happy(Bob)
\]
Probabilistic Knowledge Bases

PKB = Set of formulas and their probabilities
  + Consistency + Maximum entropy
  = Set of formulas and their weights
    (a.k.a. Markov logic network)
  = Set of formulas and their potentials
    (1 if formula true, $\phi_i$ if formula false)

\[
P(\text{world}) = \frac{1}{Z} \prod_i \phi_i^{n_i(\text{world})}
\]
Weighted Model Counting

- ModelCount(CNF) = # worlds that satisfy CNF
- Assign a weight to each literal
- Weight(world) = \( \prod \) weights(true literals)
- Weighted model counting:
  Given CNF \( C \) and literal weights \( W \)
  Output \( \sum \) weights(worlds that satisfy \( C \))

PTP is reducible to lifted WMC
Inference Problems

TP₀ = SAT

Lifted

Weighted

Counting

TP₁

LWSAT

PTP = LWMC

LMC

MPE = WSAT

PI = WMC

MC
Propositional Case

- All conditional probabilities are ratios of partition functions:

\[
P(\text{Query} \mid \text{PKB}) = \frac{\sum_{\text{worlds}} 1_{\text{Query}}(\text{world}) \prod_i \Phi_i(\text{world})}{Z(\text{P KB})}
\]

\[
= \frac{Z(\text{P KB} \cup \{(\text{Query}, 0)\})}{Z(\text{P KB})}
\]

- All partition functions can be computed by weighted model counting
Conversion to CNF + Weights

\[ \text{WCNF}(PKB) \]

\begin{align*}
&\text{for all } (F_i, \Phi_i) \in PKB \text{ s.t. } \Phi_i > 0 \text{ do} \\
&PKB \leftarrow PKB \cup \{(F_i \leftrightarrow A_i, 0)\} \setminus \{(F_i, \Phi_i)\} \\
&CNF \leftarrow \text{CNF}(PKB) \\
&\text{for all } \neg A_i \text{ literals do } W_{\neg A_i} \leftarrow \Phi_i \\
&\text{for all other literals } L \text{ do } w_L \leftarrow 1 \\
&\text{return } (CNF, \text{weights})
\end{align*}
Probabilistic Theorem Proving

\[
\text{PTP}(PKB, \text{Query})
\]

\[
PKB_Q \leftarrow PKB U \{(\text{Query}, 0)\}
\]

\[
\text{return } \frac{\text{WMC}(\text{WCNF}(PKB_Q))}{\text{WMC}(\text{WCNF}(PKB))}
\]
Probabilistic Theorem Proving

\[
\text{PTP}(PKB, \text{Query})
\]

\[
PKB_Q \leftarrow PKB \cup \{(\text{Query},0)\}
\]

\[
\text{return } \frac{\text{WMC}(\text{WCNF}(PKB_Q))}{\text{WMC}(\text{WCNF}(PKB))}
\]

Compare:

\[
\text{TP}(KB, \text{Query})
\]

\[
KB_Q \leftarrow KB \cup \{\neg \text{Query}\}
\]

\[
\text{return } \neg \text{SAT}(\text{CNF}(KB_Q))
\]
Weighted Model Counting

\[ \text{WMC}(CNF, \text{weights}) \]

\[ \text{if all clauses in } CNF \text{ are satisfied} \]
\[ \text{return } \prod_{A \in A(CNF)} (w_A + w_{\neg A}) \]

\[ \text{if } CNF \text{ has empty unsatisfied clause } \text{return } 0 \]
Weighted Model Counting

\[ \text{WM}_\text{C}(\text{CNF}, \text{weights}) \]

- **if** all clauses in CNF are satisfied
  \[ \text{return } \prod_{A \in \text{A(CNF)}}(w_A + w_{\neg A}) \]

- **if** CNF has empty unsatisfied clause \text{return } 0

- **if** CNF can be partitioned into CNFs \( C_1, \ldots, C_k \) sharing no atoms
  \[ \text{return } \prod_{i=1}^{k} \text{WM}_\text{C}(C_i, \text{weights}) \]

Decomp. Step
**Weighted Model Counting**

\[ \text{WMC}(CNF, \text{weights}) \]

- **if** all clauses in \( CNF \) are satisfied
  \[ \text{return } \prod_{A \in A(CNF)} (w_A + w_{\neg A}) \]
- **if** \( CNF \) has empty unsatisfied clause **return** 0
- **if** \( CNF \) can be partitioned into CNFs \( C_1, \ldots, C_k \) sharing no atoms
  \[ \text{return } \prod_{i=1}^{k} \text{WMC}(C_i, \text{weights}) \]
- choose an atom \( A \)
  \[ \text{return } w_A \text{WMC}(CNF \mid A, \text{weights}) + w_{\neg A} \text{WMC}(CNF \mid \neg A, \text{weights}) \]

**Splitting Step**
First-Order Case

- PTP schema remains the same
- Conversion of PKB to hard CNF and weights:
  New atom in $F_i \iff A_i$ is now $\text{Predicate}_i(\text{variables in } F_i, \text{constants in } F_i)$
- Lift each step of WMC
Lifted Weighted Model Counting

\textbf{LWMC}(CNF, substs, weights)

\begin{enumerate}
\item \textbf{if} all clauses in \textit{CNF} are satisfied
  \begin{align*}
  \text{return } \prod_{A \in A(CNF)} (w_A + w_{\neg A})^{n_A(substs)}
  \end{align*}
\item \textbf{if} \textit{CNF} has empty unsatisfied clause \textbf{return} 0
\end{enumerate}
Lifted Weighted Model Counting

\[ \text{LWMC}(CNF, \text{subs}, \text{weights}) \]

\[
\begin{align*}
&\text{if all clauses in } CNF \text{ are satisfied} \\
&\quad \text{return } \prod_{A \in A(CNF)} \left( w_A + w_{\overline{A}} \right)^{n_A(\text{subs})} \\
&\text{if } CNF \text{ has empty unsatisfied clause return } 0 \\
&\text{if there exists a lifted decomposition of } CNF \\
&\quad \text{return } \prod_{i=1}^{k} [\text{LWMC}(CNF_{i,1}, \text{weights})]^{m_i}
\end{align*}
\]
Lifted Weighted Model Counting

\( \text{LWMC}(CNF, \text{subs}, \text{weights}) \)

- If all clauses in \( CNF \) are satisfied
  - Return \( \prod_{A \in A(CNF)} (w_A + w_{\neg A})^{n_A(\text{subs})} \)
- If \( CNF \) has empty unsatisfied clause return 0
- If there exists a lifted decomposition of \( CNF \)
  - Return \( \prod_{i=1}^{k} [\text{LWMC}(CNF_{i,1}, \text{weights})]^{m_i} \)
- Choose an atom \( A \)
  - Return \( \sum_{i=1}^{l} n_i w_A^{t_i} w_{\neg A}^{f_i} \text{LWMC}(CNF | \sigma_j, \text{weights}) \)
Lifting the Decomposition and Splitting Steps: Details

- We will assume Normal forms
  - Step 1: Convert to Normal form if not in one

A WCNF is in normal form if:

- There are no constants in any formula
- If two distinct atoms with the same predicate symbol have variables \( x \) and \( y \) in the same position then \( \Delta_x = \Delta_y \) where \( \Delta_x \) and \( \Delta_y \) are the domains of \( x \) and \( y \) respectively.

Example: \((\text{Smokes}(x) \Rightarrow \text{Asthma}(x))\) and \((\text{Smokes}(\text{Ana}))\).

Normal Form: \((\text{Smokes}(x') \Rightarrow \text{Asthma}(x'))\), \((\text{Smokes1}(y) \Rightarrow \text{Asthma1}(y))\) and \((\text{Smokes1}(y))\), where \( \Delta'_x = \Delta_x \setminus \{\text{Ana}\} \) and \( \Delta_y = \{\text{Ana}\} \).
Lifting the Decomposition Step

- Propositional Decomposition: Independent
  \[ \prod_{i=1}^{n} WM C(C_i) \]

- Lifted Decomposition: Independent and Identical (symmetry)
  \[ [WM C(C_i)]^n \]

- How to find “independent and identical decompositions” by just looking at the logical structure?
  - POWER RULE (Jha-Gogate-Meliou-Suciu, 2010)
POWER RULE

- Decomposer: Given a predicate “R”, a logical variable “x” at position “i” in “R” is a decomposer if by grounding “x” to any two constants “A” and “B” in its domain yields a decomposable WCNF having two components such that
  - Both components are identical subject to renaming of predicates and constants.

- \( R(x) \lor S(x); S(y) \lor T(y) \) --- “x” is a decomposer
- \( R(x) \lor S(x); S(x) \lor T(y) \) --- No decomposer
How to use the decomposer?

- Substitute “x” with a constant in the domain.
- Return $[\text{LWMC (x}\setminus A)]^n$
  - Where $n$ is the number of constants in the domain of “x”.
Lifting the Conditioning or Splitting Step

- Singleton Rule
  - If an atom “R” is singleton, namely if the predicate symbol contains only one logical variable then splitting on it creates exactly “n+1” distinct WCNFs. Moreover, each of these WCNFs appear exactly “n choose k” times where “0<=k<=n.”
  - Examples: “R(x) v T(y)”
  - R(x) v S(x); S(x) v T(y)
How to use the Singleton Rule

- Split the WCNF into “n+1” WCNFs
- In the i-th WCNF
  - For each clause in which R appears
    - Replace the domain size of all variables which are shared with “R” by “n-i” if the literal of “R” is positive and by “i” if the literal of “R” is negative
    - Keep other domain sizes the same
  - Multiply the result by “n choose i”
Extensions

- Unit propagation, etc.
- Caching / Memoization
- Knowledge-based model construction
Formal properties

- PTP can be exponentially more efficient than FOVE (first-order variable elimination)
- PTP has the same worst-case time and space complexity as FOVE
Approximate Inference

\[ \text{WMC}(\text{CNF}, \text{weights}) \]

if all clauses in \( \text{CNF} \) are satisfied

\[
\text{return } \prod_{A \in \text{A}(\text{CNF})} (w_A + w_{\overline{A}})
\]

if \( \text{CNF} \) has empty unsatisfied clause return 0

if \( \text{CNF} \) can be partitioned into \( \text{CNFs} \) \( C_1, \ldots, C_k \) sharing no atoms

\[
\text{return } \prod_{i=1}^{k} \text{WMC}(C_i, \text{weights})
\]

choose an atom \( A \)

\[
\text{return } \frac{w_A}{Q(A \mid \text{CNF}, \text{weights})} \text{WMC}(\text{CNF} \mid A, \text{weights})
\]

with probability \( Q(A \mid \text{CNF}, \text{weights}) \), etc.
Link Prediction

![Graph showing time in seconds vs. number of objects for PTP and FOVE.](image)
Coreference (Cora)

Negative log likelihood

Time in minutes

MC-LWMC
MC-WMC
Lifted-BP
MC-SAT
Conclusion

- We need inference for first-order models
- Probabilistic theorem proving is the most powerful approach to date
- PTP is reducible to lifted weighted model counting
- A single algorithm solves a lot of problems
- Empirically very efficient
- Available in the Alchemy system (alchemy.cs.washington.edu)