Probabilistic Logic Languages

Fabrizio Riguzzi

ENDIF – University of Ferrara, Italy
fabrizio.riguzzi@unife.it
Outline

1. Probabilistic Logic Languages
2. Distribution Semantics
3. Expressive Power
4. Distribution Semantics with Function Symbols
5. Reasoning Tasks
6. Inference for PLP under DS
7. Inference with Tabling
8. Conclusions
Combining Logic and Probability

- Useful to model domains with complex and uncertain relationships among entities
- Many approaches proposed in the areas of Logic Programming, Uncertainty in AI, Machine Learning, Databases
- Logic Programming: Distribution Semantics [Sato, 1995]
- A probabilistic logic program defines a probability distribution over normal logic programs (called instances or possible worlds or simply worlds)
- The distribution is extended to a joint distribution over worlds and interpretations (or queries)
- The probability of a query is obtained from this distribution
Probabilistic Logic Programming (PLP) Languages under the Distribution Semantics

- Probabilistic Logic Programs [Dantsin, 1991]
- Probabilistic Horn Abduction [Poole, 1993], Independent Choice Logic (ICL) [Poole, 1997]
- PRISM [Sato, 1995]
- Logic Programs with Annotated Disjunctions (LPADs) [Vennekens et al., 2004]
- ProbLog [De Raedt et al., 2007]

They differ in the way they define the distribution over logic programs
sneezing(\(X\)) \leftarrow \text{flu}(\(X\), \text{flu}_\text{sneezing}(\(X\)).

sneezing(\(X\)) \leftarrow \text{hay}_\text{fever}(\(X\), \text{hay}_\text{fever}_\text{sneezing}(\(X\)).

\text{flu}(\text{bob}).

\text{hay}_\text{fever}(\text{bob}).

\text{disjoint}([[\text{flu}_\text{sneezing}(\(X\)) : 0.7, \text{null} : 0.3]]).

\text{disjoint}([[\text{hay}_\text{fever}_\text{sneezing}(\(X\)) : 0.8, \text{null} : 0.2]]).

- Distributions over facts by means of \text{disjoint} statements
- \text{null} does not appear in the body of any rule
- Worlds obtained by selecting one atom from every grounding of each disjoint statement
sneezing(X) ← flu(X), msw(flu_sneezing(X), 1).
sneezing(X) ← hay_fever(X), msw(hay_fever_sneezing(X), 1).
flu(bob).
hay_fever(bob).

values(flu_sneezing(_X), [1, 0]).
values(hay_fever_sneezing(_X), [1, 0]).
: − set_sw(flu_sneezing(_X), [0.7, 0.3]).
: − set_sw(hay_fever_sneezing(_X), [0.8, 0.2]).

- Distributions over msw facts (random switches)
- Worlds obtained by selecting one value for every grounding of each msw statement
Logic Programs with Annotated Disjunctions

\[
sneezing(X) : 0.7 \lor null : 0.3 \leftarrow flu(X).
\]
\[
sneezing(X) : 0.8 \lor null : 0.2 \leftarrow hay\_fever(X).
\]
\[
flu(bob).
\]
\[
hay\_fever(bob).
\]

- Distributions over the head of rules
- *null* does not appear in the body of any rule
- Worlds obtained by selecting one atom from the head of every grounding of each clause
\[
\text{sneezing}(X) \leftarrow \text{flu}(X), \text{flu\_sneezing}(X).
\text{sneezing}(X) \leftarrow \text{hay\_fever}(X), \text{hay\_fever\_sneezing}(X).
\text{flu}(\text{bob}).
\text{hay\_fever}(\text{bob}).
0.7 :: \text{flu\_sneezing}(X).
0.8 :: \text{hay\_fever\_sneezing}(X).
\]

- Distributions over facts
- Worlds obtained by selecting or not every grounding of each probabilistic fact
Distribution Semantics

- Case of no function symbols: finite Herbrand universe, finite set of groundings of each disjoint statement/switch/clause

  **Atomic choice**: selection of the $i$-th atom for grounding $C\theta$ of disjoint statement/switch/clause $C$
  - represented with the triple $(C, \theta, i)$
  - a ProbLog fact $p :: F$ is interpreted as $F : p \lor \text{null} : 1 - p$.

  **Example** $C_1 = \text{disjoint}([\text{flu_sneezing}(X) : 0.7, \text{null} : 0.3])$, $(C_1, \{X/bob\}, 1)$

- **Composite choice $\kappa$**: consistent set of atomic choices
  - $\kappa = \{(C_1, \{X/bob\}, 1), (C_1, \{X/bob\}, 2)\}$ not consistent
  - The probability of composite choice $\kappa$ is

$$P(\kappa) = \prod_{(C,\theta,i) \in \kappa} P_0(C, i)$$
**Selection $\sigma$:** a total composite choice (one atomic choice for every grounding of each disjoint statement/clause)

\[ \sigma = \{(C_1, \{X/bob\}, 1), (C_2, \{bob\}, 1)\} \]

- \( C_1 = \text{disjoint([flu\_sneezing(X) : 0.7, null : 0.3])} \).
- \( C_2 = \text{disjoint([hay\_fever\_sneezing(X) : 0.8, null : 0.2])} \).

A selection $\sigma$ identifies a logic program $w_\sigma$ called **world**

The probability of $w_\sigma$ is $P(w_\sigma) = P(\sigma) = \prod_{(C, \theta, i) \in \sigma} P_0(C, i)$

Finite set of worlds: $W_T = \{w_1, \ldots, w_m\}$

$P(w)$ distribution over worlds: $\sum_{w \in W_T} P(w) = 1$
Distribution Semantics

- Herbrand base $H_T = \{A_1, \ldots, A_n\}$
- Query $Q$: $P(Q|w) = 1$ if $w \models Q$ and 0 otherwise
- $P(Q) = \sum_w P(Q, w) = \sum_w P(Q|w)P(w) = \sum_{w \models Q} P(w)$
Example Program (ICL)

4 worlds

\[ \text{sneezing}(X) \leftarrow \text{flu}(X), \text{flu}_{-}\text{sneezing}(X). \]
\[ \text{sneezing}(X) \leftarrow \text{hay}_{-}\text{fever}(X), \text{hay}_{-}\text{fever}_{-}\text{sneezing}(X). \]
\[ \text{flu}(bob). \]
\[ \text{hay}_{-}\text{fever}(bob). \]
\[ \text{flu}_{-}\text{sneezing}(bob). \]
\[ \text{null}. \]
\[ \text{hay}_{-}\text{fever}_{-}\text{sneezing}(bob). \]
\[ \text{hay}_{-}\text{fever}_{-}\text{sneezing}(bob). \]
\[ P(w_1) = 0.7 \times 0.8 \]
\[ P(w_2) = 0.3 \times 0.8 \]
\[ P(w_3) = 0.7 \times 0.2 \]
\[ P(w_4) = 0.3 \times 0.2 \]

- \text{sneezing}(bob) is true in 3 worlds
- \[ P(\text{sneezing}(bob)) = 0.7 \times 0.8 + 0.3 \times 0.8 + 0.7 \times 0.2 = 0.94 \]
Example Program (LPAD)

4 worlds

\[\begin{align*}
sneezing(bob) & \leftarrow flu(bob). \\
sneezing(bob) & \leftarrow hay\_fever(bob). \\
flu(bob). & \\
hay\_fever(bob). & \\
P(w_1) & = 0.7 \times 0.8 \\
\end{align*}\]

\[\begin{align*}
sneezing(bob) & \leftarrow null \leftarrow flu(bob). \\
null & \leftarrow hay\_fever(bob). \\
flu(bob). & \\
hay\_fever(bob). & \\
P(w_2) & = 0.3 \times 0.8 \\
\end{align*}\]

\[\begin{align*}
sneezing(bob) & \leftarrow flu(bob). \\
null & \leftarrow hay\_fever(bob). \\
flu(bob). & \\
hay\_fever(bob). & \\
P(w_3) & = 0.7 \times 0.2 \\
\end{align*}\]

\[\begin{align*}
sneezing(bob) & \leftarrow null \leftarrow hay\_fever(bob). \\
null & \leftarrow hay\_fever(bob). \\
flu(bob). & \\
hay\_fever(bob). & \\
P(w_4) & = 0.3 \times 0.2 \\
\end{align*}\]

\[\begin{align*}
sneezing(bob) \text{ is true in 3 worlds} \\
P(sneezing(bob)) & = 0.7 \times 0.8 + 0.3 \times 0.8 + 0.7 \times 0.2 = 0.94 \\
\end{align*}\]
Example Program (ProbLog)

- 4 worlds

\[
\text{sneezing}(X) \leftarrow \text{flu}(X), \text{flu\_sneezing}(X).
\]
\[
\text{sneezing}(X) \leftarrow \text{hay\_fever}(X), \text{hay\_fever\_sneezing}(X).
\]
\[
\text{flu}(bob).
\]
\[
\text{hay\_fever}(bob).
\]

\[
\text{flu\_sneezing}(bob).
\]
\[
\text{hay\_fever\_sneezing}(bob).
\]
\[
P(w_1) = 0.7 \times 0.8 \quad P(w_2) = 0.3 \times 0.8
\]

\[
P(w_3) = 0.7 \times 0.2 \quad P(w_4) = 0.3 \times 0.2
\]

- \text{sneezing}(bob) is true in 3 worlds

\[
P(\text{sneezing}(bob)) = 0.7 \times 0.8 + 0.3 \times 0.8 + 0.7 \times 0.2 = 0.94
\]
Examples

Throwing coins

\[\begin{align*}
\text{heads}(\text{Coin}): 1/2 &; \quad \text{tails}(\text{Coin}): 1/2 :- \text{toss}(\text{Coin}), \neg \text{biased}(\text{Coin}). \\
\text{heads}(\text{Coin}): 0.6 &; \quad \text{tails}(\text{Coin}): 0.4 :- \text{toss}(\text{Coin}), \text{biased}(\text{Coin}). \\
\text{fair}(\text{Coin}): 0.9 &; \quad \text{biased}(\text{Coin}): 0.1. \\
\text{toss}(\text{coin}).
\end{align*}\]

Russian roulette with two guns

\[\begin{align*}
\text{death}: 1/6 :- \text{pull\_trigger}(\text{left\_gun}). \\
\text{death}: 1/6 :- \text{pull\_trigger}(\text{right\_gun}). \\
\text{pull\_trigger}(\text{left\_gun}). \\
\text{pull\_trigger}(\text{right\_gun}).
\end{align*}\]
Examples

Mendel’s inheritance rules for pea plants

color(X, white) :- cg(X, 1, w), cg(X, 2, w).
color(X, purple) :- cg(X, _A, p).
cg(X, 1, A) : 0.5 ; cg(X, 1, B) : 0.5 :-
    mother(Y, X), cg(Y, 1, A), cg(Y, 2, B).
cg(X, 2, A) : 0.5 ; cg(X, 2, B) : 0.5 :-
    father(Y, X), cg(Y, 1, A), cg(Y, 2, B).

Probability of paths

path(X, X).
path(X, Y) :- path(X, Z), edge(Z, Y).
edge(a, b) : 0.3.
edge(b, c) : 0.2.
edge(a, c) : 0.6.
Encodings of Bayesian Networks

- Burglary
- Earthquake
- Alarm

**Probabilities**

<table>
<thead>
<tr>
<th>Event</th>
<th>t</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>burg</td>
<td>0.1</td>
<td>0.9</td>
</tr>
<tr>
<td>earthq</td>
<td>0.2</td>
<td>0.8</td>
</tr>
<tr>
<td>alarm</td>
<td>1.0</td>
<td>0.0</td>
</tr>
<tr>
<td>alarm</td>
<td>0.8</td>
<td>0.2</td>
</tr>
<tr>
<td>alarm</td>
<td>0.8</td>
<td>0.2</td>
</tr>
<tr>
<td>alarm</td>
<td>0.1</td>
<td>0.9</td>
</tr>
</tbody>
</table>

**Logic Rules**

- `burg(t): 0.1 ; burg(f): 0.9.`
- `earthq(t): 0.2 ; earthq(f): 0.8.`
- `alarm(t): ¬burg(t), earthq(t).`
- `alarm(t): 0.8 ; alarm(f): 0.2: ¬burg(t), earthq(f).`
- `alarm(t): 0.8 ; alarm(f): 0.2: ¬burg(f), earthq(t).`
- `alarm(t): 0.1 ; alarm(f): 0.9: ¬burg(f), earthq(f).`
Expressive Power

- All these languages have the same expressive power
- LPADs have the most general syntax
- There are transformations that can convert each one into the others
- ICL, PRISM: direct mapping
- ICL, PRISM to LPAD: direct mapping
Clause $C_i$ with variables $\overline{X}$

\[
H_1 : p_1 \lor \ldots \lor H_n : p_n \leftarrow B.
\]

is translated into

\[
H_1 \leftarrow B, \text{choice}_{i,1}(\overline{X}).
\]
\[
\vdots
\]
\[
H_n \leftarrow B, \text{choice}_{i,1}(\overline{X}).
\]

\[
disjoint([\text{choice}_{i,1}(\overline{X}) : p_1, \ldots, \text{choice}_{i,n}(\overline{X}) : p_n]).
\]
Clause $C_i$ with variables $\bar{X}$

$$H_1 : p_1 \lor \ldots \lor H_n : p_n \leftarrow B.$$ 

is translated into

$$H_1 \leftarrow B, f_{i,1}(\bar{X}).$$
$$H_2 \leftarrow B, \text{not}(f_{i,1}(\bar{X})), f_{i,2}(\bar{X}).$$
$$\vdots$$
$$H_n \leftarrow B, \text{not}(f_{i,1}(\bar{X})), \ldots, \text{not}(f_{i,n-1}(\bar{X})).$$

$$\pi_1 :: f_{i,1}(\bar{X}).$$
$$\vdots$$
$$\pi_{n-1} :: f_{i,n-1}(\bar{X}).$$

where $\pi_1 = p_1$, $\pi_2 = \frac{p_2}{1-\pi_1}$, $\pi_3 = \frac{p_3}{(1-\pi_1)(1-\pi_2)}$, $\ldots$

In general $\pi_j = \frac{p_j}{\prod_{j=1}^{j-1}(1-\pi_j)}$, $\ldots$. 

Fabrizio Riguzzi (University of Ferrara)
Negation

- How to deal with negation?
- Each world should have a single total model because we consider two-valued interpretations
- We want to model uncertainty only by means of random choices
- This can be required explicitly: each world should have a total well founded model/single stable model (sound programs)
Function Symbols

- What if function symbols are present?
- Infinite, countable Herbrand universe
- Infinite, countable Herbrand base
- Infinite, countable grounding of the program $T$
- Uncountable $W_T$
- Each world infinite, countable
- $P(w) = 0$
- Semantics not well-defined
Game of dice

\[
\text{on}(0,1) : 1/3 ; \quad \text{on}(0,2) : 1/3 ; \quad \text{on}(0,3) : 1/3.
\]
\[
\text{on}(T,1) : 1/3 ; \quad \text{on}(T,2) : 1/3 ; \quad \text{on}(T,3) : 1/3 :-
\text{T1 is } T-1, \text{ T1}\geq 0, \text{ on}(T1,F), \text{ \textbackslash+ on}(T1,3).
\]
Hidden Markov Models

\[ X(t-1) \rightarrow Y(t-1) \rightarrow X(t) \rightarrow Y(t) \rightarrow X(t+1) \rightarrow Y(t+1) \]

\[
hmm(S, O) :- hmm(q1, [], S, O).
\]

\[
hmm(end, S, S, []). \]

\[
hmm(Q, S0, S, [L|O]) :-
Q \neq end,
next_state(Q, Q1, S0),
letter(Q, L, S0),
hmm(Q1, [Q|S0], S, O).
\]

\[
next_state(q1, q1, _S): 1/3; next_state(q1, q2, _S): 1/3;
next_state(q1, end, _S): 1/3.
\]

\[
next_state(q2, q1, _S): 1/3; next_state(q2, q2, _S): 1/3;
next_state(q2, end, _S): 1/3.
\]

\[
letter(q1, a, _S): 0.25; letter(q1, c, _S): 0.25;
letter(q1, g, _S): 0.25; letter(q1, t, _S): 0.25.
\]

\[
letter(q2, a, _S): 0.25; letter(q2, c, _S): 0.25;
letter(q2, g, _S): 0.25; letter(q2, t, _S): 0.25.
\]
Semantics proposed for ICL and PRISM, applicable also to the other languages

Definition of a probability measure $\mu$ over $W_T$

$\mu$ assign a probability to every element of an algebra $\Omega$ of subsets of $W_T$, i.e. a set of subsets closed under union and complementation

The algebra $\Omega$ is the set of sets of worlds identified by a finite set of finite composite choices
Composite Choices

- Set of worlds compatible with $\kappa$: $\omega_\kappa = \{ w_\sigma \in W_T | \kappa \subseteq \sigma \}$
- For programs without function symbols $P(\kappa) = \sum_{w \in \omega_\kappa} P(w)$

```
sneezing(X) ← flu(X), flu_sneezing(X).
sneezing(X) ← hay_fever(X), hay_fever_sneezing(X).
flu(bob).
hay_fever(bob).
C_1 = disjoint([flu_sneezing(X) : 0.7, null : 0.3]).
C_2 = disjoint([hay_fever_sneezing(X) : 0.8, null : 0.2]).
```

- $\kappa = \{(C_1, \{X/bob\}, 1)\}$, $\omega_\kappa =$
  - flu_sneezing(bob).
  - hay_fever_sneezing(bob).
- $P(w_1) = 0.7 \times 0.8$
- $P(w_2) = 0.7 \times 0.2$
- $P(\kappa) = 0.7 = P(w_1) + P(w_2)$
Sets of Composite Choices

- Set of composite choices $K$
- Set of worlds compatible with $K$: $\omega_K = \bigcup_{\kappa \in K} \omega_{\kappa}$
- $\Omega = \{ \omega_K | K \text{ is finite set of finite composite choices} \}$
- Two composite choices $\kappa_1$ and $\kappa_2$ are exclusive if their union is inconsistent

  - $\kappa_1 = \{ (C_1, \{X/bob\}, 1) \}$,
  - $\kappa_2 = \{ (C_1, \{X/bob\}, 2), (C_2, \{X/bob\}, 1) \}$
  - $\kappa_1 \cup \kappa_2$ inconsistent

- A set $K$ of composite choices is mutually exclusive if for all $\kappa_1 \in K, \kappa_2 \in K, \kappa_1 \neq \kappa_2 \Rightarrow \kappa_1$ and $\kappa_2$ are exclusive.

- If $K$ is mutually exclusive, define $P(K) = \sum_{\kappa \in K} P(\kappa)$

Lemma ([Poole, 2000])

If $K$ and $K'$ are both mutually exclusive sets of composite choices such that $\omega_K = \omega_{K'}$, then $P(K) = P(K')$
Probability Measure

Lemma ([Poole, 2000])

Given a finite set $K$ of finite composite choices, there exists a finite set $K'$ of finite composite choices that is mutually exclusive and such that $\omega_K = \omega_{K'}$. 

- $\Omega = \{\omega_K | K \text{ is a finite set of finite composite choices}\}$
- $\Omega$ is an algebra

Definition

$\mu : \Omega \rightarrow [0, 1]$ is

$$\mu(\omega) = P(K)$$

for $\omega \in \Omega$ where $K$ is a mutually exclusive finite set of finite composite choices such that $\omega_K = \omega$. 
Probability Measure

- \( \mu \) satisfies the finite additivity version of Kolmogorov probability axioms
  1. \( \mu(\omega) \geq 0 \) for all \( \omega \in \Omega \)
  2. \( \mu(W) = 1 \)
  3. \( \omega_1 \cap \omega_2 = \emptyset \rightarrow \mu(\omega_1 \cup \omega_2) = \mu(\omega_1) + \mu(\omega_2) \) for all \( \omega_1 \in \Omega, \omega_2 \in \Omega \)

- So \( \mu \) is a probability measure
Probability of a Query

- Given a query $Q$, a composite choice $\kappa$ is an explanation for $Q$ if
  \[ \forall w \in \omega_\kappa \quad w \models Q \]
- A set $K$ of composite choices is covering wrt $Q$ if every world in which $Q$ is true belongs to $\omega_K$

Definition

\[ P(Q) = \mu(\{ w | w \in W_T, w \models Q \}) \]

- If $Q$ has a finite set of finite explanations that is covering, $P(Q)$ is well-defined
Example Program (ICL)

\[
\begin{align*}
\text{sneezing}(X) & \leftarrow \text{flu}(X), \text{flu\_sneezing}(X). \\
\text{sneezing}(X) & \leftarrow \text{hay\_fever}(X), \text{hay\_fever\_sneezing}(X). \\
\text{flu}(bob). \\
\text{hay\_fever}(bob). \\
C_1 & = \text{disjoint}([\text{flu\_sneezing}(X) : 0.7, \text{null} : 0.3]). \\
C_2 & = \text{disjoint}([\text{hay\_fever\_sneezing}(X) : 0.8, \text{null} : 0.2]).
\end{align*}
\]

- **Goal** **\text{sneezing}(bob)**
- \(\kappa_1 = \{(C_1, \{X/bob\}, 1)\}\)
- \(\kappa_2 = \{(C_1, \{X/bob\}, 2), (C_2, \{X/bob\}, 1)\}\)
- \(K = \{\kappa_1, \kappa_2\}\) mutually exclusive finite set of finite explanations that are covering for **\text{sneezing}(bob)**
- \(P(Q) = P(\kappa_1) + P(\kappa_2) = 0.7 + 0.3 \cdot 0.8 = 0.94\)
Reasoning Tasks

- Inference: we want to compute the probability or an explanation of a query given the model and, possibly, some evidence.
- Weight learning: we know the structural part of the model (the logic formulas) but not the numeric part (the weights) and we want to infer the weights from data.
- Structure learning: we want to infer both the structure and the weights of the model from data.
Inference Tasks

- Computing the (conditional) probability of a ground query given the model and, possibly, some evidence
- Finding the most likely state of a set of query atoms given the evidence (Maximum A Posteriori/Most Probable Explanation inference)
  - In Hidden Markov Models, the most likely state of the state variables given the observations is the Viterbi path, its probability the Viterbi probability
- Finding the \((k)\) most probable explanation(s)
- Finding the distribution of variable substitutions for a non-ground query.
- Finding the most probable variable substitution for a non-ground query.
Weight Learning

- **Given**
  - model: a probabilistic logic model with unknown parameters
  - data: a set of interpretations

- **Find the values of the parameters that maximize the probability of the data given the model**

- **Discriminative learning:** maximize the conditional probability of a set of outputs (e.g. ground instances for a predicate) given a set of inputs

- **Alternatively, the data are queries for which we know the probability:** minimize the error in the probability of the queries that is returned by the model
Structure Learning

- Given
  - language bias: a specification of the search space
  - data: a set of interpretations
- Find the formulas and the parameters that maximize the likelihood of the data given the model
- Discriminative learning: again maximize the conditional likelihood of a set of outputs given a set of inputs
Computing the probability of a query (no evidence)

Explanation based:
- find explanations for queries
- make the explanations mutually exclusive
  - by means of an iterative splitting algorithm (Ailog2 [Poole, 2000])
  - by means of Binary Decision Diagrams (ProbLog [De Raedt et al., 2007], cplint [Riguzzi, 2007, Riguzzi, 2009] PITA [Riguzzi and Swift, 2010])

Bayesian Network based:
- Convert to BN
- Use BN inference algorithms (CVE [Meert et al., 2009])
- Lifted inference
sneezing(X) ← flu(X), flu_sneezing(X).
sneezing(X) ← hay_fever(X), hay_fever_sneezing(X).
flu(david).
hay_fever(david).
\( C_1 = 0.7 \) :: flu_sneezing(X).
\( C_2 = 0.8 \) :: hay_fever_sneezing(X).

- Distributions over facts
Finding Explanations

- All explanations for the query are collected
- ProbLog: source to source transformation for facts, use of dynamic database
- cplint: meta-interpretation
- PITA: source to source transformation, addition of an argument to predicates
Explanation Based Inference Algorithm

- $K =$ set of explanations found for $Q$,
- They are not necessarily mutually exclusive
- The probability of $Q$ is given by the probability of the formula

$$f_K(Y) = \bigvee_{\kappa \in K} \bigwedge_{(C,\theta,i) \in \kappa} (Y_{C\theta} = i)$$

where $Y_{C\theta}$ is a random variable whose domain is 1, 2 and $P(Y_{C\theta} = i) = P_0(C, i)$
- Binary domain: we use a Boolean variable $X_{C\theta}$ to represent ($Y_{C\theta} = 1$)
- $\neg X_{C\theta}$ represents ($Y_{C\theta} = 2$)
Example

A set of covering explanations for *sneezing*(*david*) is $K = \{\kappa_1, \kappa_2\}$

$\kappa_1 = \{(C_1, \{X/david\}, 1)\}$

$\kappa_2 = \{(C_2, \{X/david\}, 1)\}$

$K = \{\kappa_1, \kappa_2\}$

$f_K(Y) = (Y_{C_1}\{X/david\} = 1) \lor (Y_{C_1}\{X/david\} = 1)$.

$X_1 = (Y_{C_1}\{X/david\} = 1)$

$X_2 = (Y_{C_2}\{X/david\} = 1)$

$f_K(X) = X_1 \lor X_2$.

$P(f_K(X)) = P(X_1 \lor X_2)$

$P(f_K(X)) = P(X_1) + P(X_2) - P(X_1)P(X_2)$

- In order to compute the probability, we must make the explanations mutually exclusive

- [De Raedt et al., 2007]: Binary Decision Diagram (BDD)
Binary Decision Diagrams

\[ f_K(\mathbf{X}) = X_1 \times f_K^{X_1}(\mathbf{X}) + \neg X_1 \times f_K^{\neg X_1}(\mathbf{X}) \]

\[ P(f_K(\mathbf{X})) = P(X_1)P(f_K^{X_1}(\mathbf{X})) + (1 - P(X_1))P(f_K^{\neg X_1}(\mathbf{X})) \]

\[ P(f_K(\mathbf{X})) = 0.7 \cdot P(f_K^{X_1}(\mathbf{X})) + 0.3 \cdot P(f_K^{\neg X_1}(\mathbf{X})) \]
Probability from a BDD

Dynamic programming algorithm [De Raedt et al., 2007]

1: function \( \text{PROB}(n) \)
2: if \( n \) is a terminal note then
3: return \( \text{value}(n) \)
4: else
5: return \( \text{PROB}(\text{child}_0(n)) \times p(v(n)) + \text{PROB}(\text{child}_1(n)) \times (1 - p(v(\text{node}))) \)
6: end if
7: end function
Logic Programs with Annotated Disjunctions

\[ C_1 = \text{strong\_sneezing}(X) : 0.3 \lor \text{moderate\_sneezing}(X) : 0.5 \leftarrow \text{flu}(X). \]
\[ C_2 = \text{strong\_sneezing}(X) : 0.2 \lor \text{moderate\_sneezing}(X) : 0.6 \leftarrow \text{hay\_fever}(X). \]
\[ C_3 = \text{flu}(\text{david}). \]
\[ C_4 = \text{hay\_fever}(\text{david}). \]

- More than two head atoms
Example

A set of covering explanations for *strong_sneezing(david)* is

\[ K = \{ \kappa_1, \kappa_2 \} \]

\[ \kappa_1 = \{ (C_1, \{ X/david \}, 1) \} \]

\[ \kappa_2 = \{ (C_2, \{ X/david \}, 1) \} \]

\[ K = \{ \kappa_1, \kappa_2 \} \]

\[ X_1 = X_{C_1}\{ X/david \} \]

\[ X_2 = X_{C_2}\{ X/david \} \]

\[ f_K(X) = (X_1 = 1) \lor (X_2 = 1). \]

\[ P(f_X) = P(X_1 = 1) + P(X_2 = 1) - P(X_1 = 1)P(X_2 = 1) \]

- To make the explanations mutually exclusive: Multivalued Decision Decision Diagram (MDD)
- Converted to BDD using a transformation similar to LPAD to ProbLog
Tabling

- PITA (Probabilistic Inference with Tabling and Answer subsumption) [Riguzzi and Swift, 2010, Riguzzi and Swift, 2011] (a package of XSB)
- All the explanations for a goal have to be found
- It makes sense to store the explanations for subgoals with tabling
- Associate to each answer (ground atom) a BDD representing its explanations
- Combine BDDs by using the Boolean operators offered by BDD manipulating packages
- Library for manipulating BDD directly in Prolog (interface to CUDD)
- A BDD is represented in Prolog by an integer indicating the address of its root node
- Casting for integer-pointer conversion
Library Predicates

- **init, end**: for allocation and deallocation of a BDD manager
- **zero(-BDD), one(-BDD), and (+BDD1, +BDD2, -BDD0)**, **or (+BDD1, +BDD2, -BDD0)**, **not (+BDD1, -BDD0)**: BDD operations
- **get_var_n(+R, +S, +Probs, -Var)**: returns a ground rule multi-valued random variable
- **equality(+Var, +Value, -BDD)**: BDD represents Var=Value
- **ret_prob(+BDD, -P)**: returns the probability of the formula encoded by BDD
Add an extra argument to each atom for storing a BDD

When an answer $p(x, bdd)$ is found, $bdd$ represents the explanations for $p(x)$

If the program is range restricted, $p(x)$ is ground

Use program transformation to obtain a Prolog program from an LPAD
Answer Subsumption

- Use a lattice on terms to combine different answers for the same goal
- The bottom element and the join operator of the lattice have to be specified in the tabling directives
- E.g: `:-table path(X,Y,or/3-zero/1)` means that, if two answers `path(a,b,bdd0)` and `path(a,b,bdd1)` are found, the single answer `path(a,b,bdd)` will be stored in the table where `or(bdd0,bdd1,bdd)`
Inference with Tabling

Program Transformation

- $PITA(p(a, b, c)) = p(a, b, c, D)$

The disjunctive clause

$C_r = H_1 : \alpha_1 \lor \ldots \lor H_n : \alpha_n \leftarrow L_1, \ldots, L_m.$

is transformed into the set of clauses $PITA(C_r)$

$PITA(C_r, i) = PITA(H_1) \leftarrow one(BB_0),$

$PITA(L_1), and(BB_0, B_1, BB_1),$

$\ldots,$

$PITA(L_m), and(BB_{m-1}, B_m, BB_m),$

$get\_var\_n(r, VC, [\alpha_1, \ldots, \alpha_n], Var),$

$equality(Var, i, BB),$

$and(BB_m, BB, BDD).$
Example

Clause

\[ \text{strong\_sneezing}(X) : 0.3 \lor \text{moderate\_sneezing}(X) : 0.5 \leftarrow \text{flu}(X). \]

is translated into

\[ \text{strong\_sneezing}(X, BDD) \leftarrow \]

\[ \text{one}(BB_0), \]
\[ \text{flu}(X, B_1), \text{and}(BB_0, B_1, BB_1), \]
\[ \text{get\_var\_n}(1, [X], [0.3, 0.5, 0.2], \text{Var}), \]
\[ \text{equality}(\text{Var}, 1, BB), \]
\[ \text{and}(BB_1, BB, BDD). \]

\[ \text{moderate\_sneezing}(X, BDD) \leftarrow \]

\[ \text{one}(BB_0), \]
\[ \text{flu}(X, B_1), \text{and}(BB_0, B_1, BB_1), \]
\[ \text{get\_var\_n}(1, [X], [0.3, 0.5, 0.2], \text{Var}), \]
\[ \text{equality}(\text{Var}, 2, BB), \]
\[ \text{and}(BB_1, BB, BDD). \]
Query: \textit{sneezing}(\textit{bob})

\[\leftarrow \text{init},\]
\[\text{sneezing}(\textit{bob}, \textit{BDD}),\]
\[\text{ret\_prob}(\textit{BDD}, \textit{P}),\]
\[\text{end}.\]
Experiments

- Biomine network: network of biological concepts
- Each edge has a probability
- Dataset from [De Raedt et al., 2007]: 50 sampled subnetworks of size 200, 400, ..., 10000 edges
- Sampling repeated 10 times
- Linux PCs with Intel Core 2 Duo E6550 (2,333 MHz) and 4 GB of RAM
- Execution stopped after 24 hours

```
path(X,Y) :- path(X,Y,[X],Z).
pred(X,Y,V,[Y|V]) :- arc(X,Y).
pred(X,Y,V0,V1) :- arc(X,Z),append(V0,_S,V1),
    \+ member(Z,V0),path(Z,Y,[Z|V0],V1).
arc(X,Y):-edge(X,Y).
arc(X,Y):-edge(Y,X).
edge('EntrezProtein_33339674','HGNC_620'):0.515062.
```
Inference with Tabling

Dataset from [De Raedt et al., 2007]

Number of solved subgraphs

Average time

<table>
<thead>
<tr>
<th>Answers</th>
<th>cplint</th>
<th>ProbLog</th>
<th>PITA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edges</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>500</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1500</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2500</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3000</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Time (s)</th>
<th>cplint</th>
<th>ProbLog</th>
<th>PITA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>500</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1500</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2500</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3000</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Game of dice

\[
on(0,1) : 1/3 \ ; \ on(0,2) : 1/3 \ ; \ on(0,3) : 1/3.
on(T,1) : 1/3 \ ; \ on(T,2) : 1/3 \ ; \ on(T,3) : 1/3 :-
T1 is T-1, T1>=0, on(T1,F), \ + \ on(T1,3).
\]
Inference with Tabling

Blood Type [Meert et al., 2009]

\[
\text{mchrom}(\text{Person}, \text{a}): 0.90 \quad \text{mchrom}(\text{Person}, \text{b}): 0.05 \quad \text{mchrom}(\text{Person}, \text{null}): 0.05 \quad :- \\
\quad \text{mother}(\text{Mother}, \text{Person}), \text{pchrom}(\text{Mother}, \text{a}), \text{mchrom}(\text{Mother}, \text{a}).
\]

\[
\text{mchrom}(\text{Person}, \text{a}): 0.49 \quad \text{mchrom}(\text{Person}, \text{b}): 0.49 \quad \text{mchrom}(\text{Person}, \text{null}): 0.02 \quad :- \\
\quad \text{mother}(\text{Mother}, \text{Person}), \text{pchrom}(\text{Mother}, \text{b}), \text{mchrom}(\text{Mother}, \text{a}).
\]

\[
\text{pchrom}(\text{Person}, \text{a}): 0.90 \quad \text{pchrom}(\text{Person}, \text{b}): 0.05 \quad \text{pchrom}(\text{Person}, \text{null}): 0.05 \quad :- \\
\quad \text{father}(\text{Father}, \text{Person}), \text{pchrom}(\text{Father}, \text{a}), \text{mchrom}(\text{Father}, \text{a}).
\]

\[
\text{bloodtype}(\text{Person}, \text{a}): 0.90 \quad \text{bloodtype}(\text{Person}, \text{b}): 0.03 \quad \text{bloodtype}(\text{Person}, \text{ab}): 0.03 \quad \\
\text{bloodtype}(\text{Person}, \text{null}): 0.04 \quad :- \quad \text{pchrom}(\text{Person}, \text{a}), \text{mchrom}(\text{Person}, \text{a}).
\]

\[
\text{bloodtype}(\text{Person}, \text{a}): 0.03 \quad \text{bloodtype}(\text{Person}, \text{b}): 0.03 \quad \text{bloodtype}(\text{Person}, \text{ab}): 0.90 \quad \\
\text{bloodtype}(\text{Person}, \text{null}): 0.04 \quad :- \quad \text{pchrom}(\text{Person}, \text{b}), \text{mchrom}(\text{Person}, \text{a}).
\]

![Graph showing time (s) vs. N with different algorithms: cplint, CVE, ProbLog, PITA]
Growing negated body [Meert et al., 2009]

\[
\begin{align*}
a0:0.5 &\quad :- \quad a1. \\
a0:0.5 &\quad :- \quad \neg a1, \quad a2. \\
a0:0.5 &\quad :- \quad \neg a1, \neg a2, \quad a3. \\
a1:0.5 &\quad :- \quad a2. \\
a1:0.5 &\quad :- \quad \neg a2, \quad a3. \\
a2:0.5 &\quad :- \quad a3. \\
a3:0.5 &
\end{align*}
\]
Growing head [Meert et al., 2009]

\[
\begin{align*}
  a_0 & : \neg a_1. \\
  a_1 & : 0.5. \\
  a_0 & : 0.5; a_1 & : 0.5 \ :- \ a_2. \\
  a_2 & : 0.5. \\
  a_0 & : 0.333333; a_1 & : 0.333333; a_2 & : 0.333333 \ :- \ a_3. \\
  a_3 & : 0.5. 
\end{align*}
\]
course(c1).
professor(p1).
student(s1).
advised_by(A,B):0.10708782742681 :- student(A),professor(B),
    position(B,faculty).
advised_by(A,B):0.0278422273781903 :- student(A),professor(B),
    \+ position(B,faculty).
course_level(A,level_300):0.0666666666666667;
course_level(A,level_400):0.318518518518519;
course_level(A,level_500):0.614814814814815 :-
    course(A).
Approximate Inference

- Inference problem is \#P hard
- For large models inference is intractable
- Approximate inference
  - Monte Carlo: draw samples of the truth value of the query
  - Iterative deepening: gives a lower and an upper bound
  - Compute only the best $k$ explanations: branch and bound, gives a lower bound
Conclusions

- Probabilistic Logic Programming: Distribution semantics
- ICL, PRISM, LPADs, ProbLog
- Expressive power
- Reasoning tasks

Thank you!
Questions?
References I


References II


