Distribution semantics and cyclic relational modeling

Taisuke Sato
Joint work with Yoshitaka Kameya & Ryosuke Kojima
New AI is coming

- Triggered by deep learning, data-centered
  - Big data (sensor data, SNS data, Web, Wikipedia, Knowledge Graph,...)
  - Machine learning (ML) techniques to deal with huge and noisy data

- Steady progress
  - Probabilistic logic learning (PLL) and statistical relational learning (SRL)
    - Probabilistic ILP that lifts propositional graphical models to first order ones
    - New probabilistic modeling languages
  - Relation extraction in IE ➔ millions of propositions from the web
  - Vector representation in NLP ➔ continuous real-valued logic

- What is required?
  - Framework unifying logic and probability (S.Russell 2013)
  - Declarative interface between modeling and ML tasks
  - Distribution semantics and PRISM are an example
From logic and probability to machine learning

- Logic
  - Completeness theorem 1930
  - Foundations of the Theory of Probability 1933

- Probability
  - Probability distribution on possible worlds

- Machine learning
  - Structure learning
  - Parameter learning
  - Viterbi inference
  - Bayesian inference

- Distribution semantics
  - Propositionalized probability computation
  - PRISM
  - Tabling(DP)

- Logic-based probabilistic modeling

Fenstad’s representation theorem 1967

ILP 2015, Kyoto
Outline

- Representation theorem
- Distribution semantics
- The PRISM language
- Infinite prob. computation for cyclic relations
- Conclusion
Let

\( L: \) countable 1st order language w.o. ”=”

\( \alpha: \) (possibly open) formula in \( L \)

\( P(\alpha): \) probability satisfying:

\[
0 \leq P(\alpha) \leq 1, P(\neg \alpha) = 1 - P(\alpha), \\
P(\alpha \lor \beta) + P(\alpha \land \beta) = P(\alpha) + P(\beta)
\]

\[
\begin{cases} 
P(\alpha) = P(\beta) & \text{if } \vdash \alpha \leftrightarrow \beta \\
P(\alpha) = 1 & \text{if } \vdash \alpha
\end{cases}
\]

Then

\[
P(\alpha) = \int_{S} P_{\omega}(\alpha[\omega])dP(\omega)
\]

where

\( S: \) set of all \( \text{H-interpretations } \omega \) on the \( \text{H-universe } D \) for \( L' \)

\( L': \) \( L \) augmented with ”special constants”

\( P(\cdot): \) \( \sigma \)-additive prob. measure on \( S \)

\( P_{\omega}(\cdot): \) prob. measure on the set of assignments \( a \in [I \rightarrow D] \)

\( I: \) variables are indexed by \( I \) (= nat numbers) like \( x_1, x_2, \ldots \)

\( \alpha[\omega]: \) \( \{ a \mid \omega \models_a \alpha(a(x_1), a(x_2), \ldots) \} = \text{set of assignments } a \text{ satisfying } \alpha \)
P(·) and P_ω(·)

- Two types of prob. measure in \( P(\alpha) = \int_S P_\omega(\alpha[\omega])dP(\omega) \)
  \( P(\exists x \text{UFO}(x)) \) vs \( P_\omega(\text{UFO}(x)) \)

- \( P(\cdot) \) gives possible worlds semantics:
  For closed \( \alpha \), \( P(\alpha) \) is the sum of probabilities of H-models \( \omega \) of \( \alpha \)
  because \( P_\omega(\alpha[\omega]) = 1 \) if \( \omega \models \alpha \), else = 0, so
  \( P(\alpha) = \int_{\omega: \omega \models \alpha} 1dP(\omega) \)

- \( P_\omega(\cdot) \) gives frequency semantics in each \( \omega \):
  For open \( \alpha = \text{Friend}(x_1, x_2) \) with two variables \( x_1, x_2 \),
  \( \text{Friend}(x_1, x_2)[\omega] \) is the set of assignments “a” s.t. \( \omega \models \text{Friend}(a(x_1), a(x_2)) \)
  \( \Rightarrow \text{Friend}(x_1, x_2)[\omega] = \{ (d_1, d_2) \text{ in } D \times D \mid \omega \models \text{Friend}(d_1, d_2) \} \)
  \( \Rightarrow P_\omega(\text{Friend}(x_1, x_2)[\omega]) = P_\omega(\{ (d_1, d_2) \text{ in } D \times D \mid \omega \models \text{Friend}(d_1, d_2) \}) \)
  = probability that randomly sampled \( (d_1, d_2) \) from \( \omega \) are friends
Simulating  \[ P(\alpha) = \int_S P_\omega(\alpha[\omega]) dP(\omega) \]

<table>
<thead>
<tr>
<th>S</th>
<th>Fr(a,a)</th>
<th>Fr(a,b)</th>
<th>Fr(b,a)</th>
<th>Fr(b,b)</th>
<th>Prob.</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\omega_1)</td>
<td>(f)</td>
<td>(f)</td>
<td>(f)</td>
<td>(t)</td>
<td>0.2</td>
</tr>
<tr>
<td>(\omega_2)</td>
<td>(t)</td>
<td>(f)</td>
<td>(t)</td>
<td>(f)</td>
<td>0.3</td>
</tr>
<tr>
<td>(\omega_3)</td>
<td>(f)</td>
<td>(t)</td>
<td>(f)</td>
<td>(f)</td>
<td>0.5</td>
</tr>
</tbody>
</table>

Other worlds have prob. 0

<table>
<thead>
<tr>
<th>((x_1,x_2))</th>
<th>Prob.</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\omega_1, \omega_2, \omega_3)</td>
<td></td>
</tr>
<tr>
<td>((a,a))</td>
<td>0.3</td>
</tr>
<tr>
<td>((a,b))</td>
<td>0.1</td>
</tr>
<tr>
<td>((b,a))</td>
<td>0.2</td>
</tr>
<tr>
<td>((b,b))</td>
<td>0.4</td>
</tr>
</tbody>
</table>

- Herbrand universe = \(D = \{a,b\}\) \(\rightarrow\) Hebrand base = \(\{Fr(a,a), Fr(a,b), Fr(b,a), Fr(b,b)\}\)
- 16 possible worlds = \(\{\omega_1, \omega_2, \omega_3, ..., \omega_{16}\}\) = truth assignments for ground atoms
- Table(left): probability of each possible world
- Table(right): probability distribution on \((x_1,x_2)\) from DxD

\[
\begin{align*}
P(\exists x_2 Fr(a,x_2)) &= P(\omega_1)*0 + P(\omega_2)*1 + P(\omega_3)*1 = 0.2*0 + 0.3*1 + 0.5*1 = 0.8 \\
P(\forall x_2 Fr(a,x_2)) &= 0 \\
P_{\omega_2}(Fr(a,x_2)) &= P_{\omega_2}(\{(a,a), (b,a)\}) = 0.3+0.2 = 0.5 \\
P(Fr(a,x_2)) &= P(\omega_1)*P_{\omega_1}(R(a,x_2)) + \cdots = 0.2*0 + 0.3*0.5 + 0.5*0.5 = 0.4
\end{align*}
\]
From logic and probability to machine learning

Logic
- Completeness theorem 1930
- Foundations of the Theory of Probability 1933

Probability

Fenstad's representation theorem 1967

possible to simulate in a computer

Logic-based probabilistic modeling

Cyclic relations
- Infinite sum of probs.
- MPE for cyclic models
- EM learning for cyclic models
- ...

Distribution semantics

Propositionalized probability computation

PRISM

Tabling(DP)

ILP 2015, Kyoto
Distribution semantics [Sato’95]

- Fenstad’s theorem (possible worlds semantics, not frequency one) adapted to probabilistic logic programs
- Program DB = F U R
  - F: ground atoms representing random choices
  - PF(.): basic distribution = distribution on H-interpretations for F
  - R: definite clauses
- PF(.) \(\Rightarrow\) least model semantics + Kolmogorov’s extension theorem
  \(\Rightarrow\) P_{DB}(.) on H-interpretations for DB
  - Least model semantics can be replaced by other semantics: stable model semantics (P-log), largest model semantics
- Semantics for PLP languages (PRISM, ProbLog, LPADs, PITA,...)
  - Infinite models such as probabilistic grammars allowed
**Propositional example**

\[ DB = \{ a:-b, a:-c, b, c \} \]

- **R**
- **F**

\[ \text{P}_F(b,c) \text{ given} \]

<table>
<thead>
<tr>
<th>( (b,c) \sim \text{P}_F(\cdot, \cdot) )</th>
<th>Sampled DB'</th>
<th>DB's least H-model</th>
<th>a</th>
<th>( P_{\text{DB}}(a,b,c) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 (false) 0</td>
<td>a:-b, a:-c</td>
<td>{}</td>
<td>0</td>
<td>0.4</td>
</tr>
<tr>
<td>0 1 (true)</td>
<td>a:-b, a:-c</td>
<td>{c,a}</td>
<td>1</td>
<td>0.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>c</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 0</td>
<td>a:-b, a:-c</td>
<td>{b,a}</td>
<td>1</td>
<td>0.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>b</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 1</td>
<td>a:-b, a:-c</td>
<td>{b,c,a}</td>
<td>1</td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>b, c</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>anything</td>
<td>else</td>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>

**P_F(0,0) | P_F(0,1) | P_F(1,0) | P_F(1,1) |
-----------|----------|----------|----------|
 0.4       | 0.3      | 0.2      | 0.1      |

*ILP 2015, Kyoto*
An implementation of the distribution semantics
Prolog + msw/2 (random choice) + parameter learning
Single data structure (explanation graph) combined with
generic algorithms subsumes:
- BNs: belief propagation algorithm
- HMMs: the Baum-Welch algorithm
- PCFGs: the Inside-Outside algorithm

Rich built-ins (applicable to all programs)
- Exact probability computation using DP on explanation graphs
- N-Viterbi (MPE) inference
- Parameter learning (EM/MAP, DAEM, VB, VT, VB-VT)
- Model scores (BIC, Cheeseman-Stutz, VFE)
- MCMC for Bayesian inference...
PRISM2.2: new features

- PRISM (http://rjida.meijo-u.ac.jp/prism/)
  - Declarative interface between probabilistic modeling and computation/learning

- PRISM2.2
  - can compute an infinite sum of probabilities for cyclic relations
    - Markov chains, prefix and infix probabilities in PCFGs
  - can learn and compute generative conditional random fields (G-CRFs)
    - logistic regression, linear-chain CRFs, CRF-CFGs
Probability computation in PRISM

- Goal → explanation graph → probability

PCFG program

```
PCFG1
S → a:0.5 | b:0.3 | S S:0.2
```

values(s,[[a],[b],[s,s]],
  set@[0.5,0.3,0.2]).

pcfg(L):- pcfg([s],L,[]).
pcfg([A|R],L0,L2):-
  ( nonterminal(A) →
    msw(A,RHS),
    pcfg(RHS,L0,L1)
  ; L0=[A|L1] ),
  pcfg(R,L1,L2).
pcfg([],L,L).

?- prob(pcfg([s],[a,b],[]),P)

Tabled search

```
pcfg([s],[a,b],[])
  ⇔ pcfg([s,s],[a,b],[[]]) & pcfg([[]],[[]],[]) & msw(s,[s,s])
  pcfg([s,s],[a,b],[[]])
  ⇔ pcfg([a],[a,b],[b]) & pcfg([s],[b],[[]]) & msw(s,[a])
  pcfg([s],[b],[[]])
  ⇔ pcfg([b],[b],[[]]) & pcfg([[]],[[]],[]) & msw(s,[b])
  pcfg([b],[b],[[]]) ⇔ pcfg([[]],[[],[]])
  pcfg([[]],[[]],[])
  pcfg([a],[a,b],[b]) ⇔ pcfg([[]],[b],[b])
pcfg([[]],[b],[b])
```

\[ P = 0.5 \times 0.3 \times 0.2 = 0.03 \]

Infinite recursion stopped!
Infinite sum of probabilities

- **Prefix** $u$: uw is a sentence for some w
- $P_{\text{prefix}}(u) = \sum_w P_{\text{cfg}}(uw)$, useful for predicting the next word
- PCFG$_1$
  
  $$S \rightarrow a:0.5 \mid b:0.3 \mid S S:0.2$$

$$P_{\text{cfg}}([a,b]) = \begin{pmatrix} \text{ } & \text{ } & \text{0.2} & \text{0.5} & \text{0.3} \\ S & S & a & b \end{pmatrix} = P(S \rightarrow S S)P(S \rightarrow a)P(S \rightarrow b)$$

$$P_{\text{prefix}}([a,b]) = \begin{pmatrix} \text{ } & \text{0.03} \\ S & a & b \end{pmatrix} + \begin{pmatrix} \text{ } & \text{0.0108} \\ S & a & b \end{pmatrix} + \cdots$$

Generated by cyclic left-corner relation of $S$ in $S \rightarrow S S$
Cyclic explanation graph

- Goal ➔ expl. graph ➔ SCCs ➔ eqs ➔ probs

PCFG$_2$

\[ S \rightarrow a:0.4 \mid b:0.3 \mid S S:0.2 \mid S:0.1 \]

values(s,[[a],[b],[s,s],[s]],
       set@[0.4,0.3,0.2,0.1]).
pre_pcfg(L):-
   pre_pcfg([s],L,[]).
pre_pcfg([A|R],L0,L2):-
   ( nonterminal(A) ➔
     msw(A,RHS),
     pre_pcfg(RHS,L0,L1)
   ; L0=[A | L1] ),
   ( L1=[] ➔ L2=[]
   ; pre_pcfg(R,L1,L2) ).
pre_pcfg([],L,L).

prefix parser program

?- lin_prob(pre_pcfg([s],[a,b],[]),P)

Tabled search

Solving prob. equations

P = 0.05

ILP 2015, Kyoto
SCCs (strongly connected components)

- SCCs are partially ordered \(\Rightarrow\) DP possible

\[
\begin{align*}
\text{pre}_{\text{pcfg}}([s],[a,b],[]) & \iff \text{pre}_{\text{pcfg}}([s],[a,b],[]) \land \text{msw}(s,[s]) \lor \text{pre}_{\text{pcfg}}([s],[a,b],[]) \land \text{msw}(s,[s]) \\
\text{pre}_{\text{pcfg}}([s],[a,b],[]) & \iff \text{pre}_{\text{pcfg}}([a],[a,b],[b]) \land \text{pre}_{\text{pcfg}}([s],[b],[]) \land \text{msw}(s,[a]) \\
& \lor \text{pre}_{\text{pcfg}}([s],[a,b],[]) \land \text{pre}_{\text{pcfg}}([s],[b],[]) \land \text{msw}(s,[s]) \\
& \lor \text{pre}_{\text{pcfg}}([s],[a,b],[]) \land \text{msw}(s,[s]) \\
\text{pre}_{\text{pcfg}}([s],[b],[]) & \iff \text{pre}_{\text{pcfg}}([b],[b],[]) \land \text{msw}(s,[b]) \lor \text{pre}_{\text{pcfg}}([s],[b],[]) \land \text{msw}(s,[s]) \\
& \lor \text{pre}_{\text{pcfg}}([s],[b],[]) \land \text{msw}(s,[s]) \\
\text{pre}_{\text{pcfg}}([a],[a,b],[b]) & \iff \text{pre}_{\text{pcfg}}([a],[a,b],[b]) \land \text{pre}_{\text{pcfg}}([s],[b],[]) \land \text{msw}(s,[a]) \\
& \lor \text{pre}_{\text{pcfg}}([s],[a,b],[b]) \land \text{pre}_{\text{pcfg}}([s],[b],[b]) \land \text{msw}(s,[s]) \\
& \lor \text{pre}_{\text{pcfg}}([a],[a,b],[b]) \iff \text{pre}_{\text{pcfg}}([s],[a,b],[b]) \\
& \lor \text{pre}_{\text{pcfg}}([s],[b],[b]) \\
\end{align*}
\]
A set of probability equations

\[
\begin{align*}
x_1 &= 0.1x_1 + 0.2x_2 \\
x_2 &= 0.1x_1 + 0.2x_2 + 0.1x_3x_6 + 0.4x_3x_7 \\
x_3 &= 0.1x_3 + 0.2x_4 + 0.3x_5 \\
x_4 &= 0.1x_3 + 0.2x_4 + 0.3x_5 \\
x_5 &= 1.0 \\
x_6 &= 0.1x_6x_8 + 0.4x_7x_8 \\
x_7 &= 1.0x_8 \\
x_8 &= 1.0
\end{align*}
\]
Plan recognition by parsing

[Vilain 90, Kautz+ 91]

Plan recognition

CFG parsing

parse tree

word

sentence

online shopper’s actions

Visit “book A” page

Visit “book B” page

Buy “book B”

S

plan

goal

(sub)goal

NT (non-terminal)

plan recognition

action seq.

successful action

plan

sentence

(1) (2) (3)
Online plan recognition

Basic idea:
Parse partial obs. as a prefix in a PCFG and compute the most likely goal and its most likely partial plan

partial observation = prefix

most likely goal

most likely partial plan

Search

Visit “book A” page

Visit “book B” page

ILP 2015, Kyoto
Classifying websurfers

Actions on a web site directory

- up : climb up
- down : down
- sibling : visit sibling page
- revisit : same page
- move : others

We capture intentions (survey, news,...) and plans of visitors **online** from their partial action sequences as a **prefix**

Websurfers climb up and down, visit sibling pages, revisit the same page, or move to other pages.

Complete log data

ILP 2015, Kyoto
Experiment

- Three data sets and five goals (survey, news, ...)
  - Clustering web log data in the Internet Traffic Archive (NASA, ClarkNet, U of S) using a mixture of PCFGs that share the same CFG but with a different set of parameters yields five clusters

- We write 102 CFG rules (32 NTs)
  - \( S \rightarrow \text{Survey}, \ S \rightarrow \text{News}, ..., \ \text{UpDown} \rightarrow \text{Up Down} \)
  - \( \text{UpDown} \rightarrow \text{Up SameLayer Down}, \ \text{Up} \rightarrow \text{Up up}, ... \)

- Parameters are learned from each dataset by MLE
  - \( \text{PCFG}_{\text{NASA}}, \ \text{PCFG}_{\text{CNet}}, \ \text{PCFG}_{\text{UofS}} \) obtained

- Complete dataset
  - Access log data paired with their most likely goal/plan inferred by the Viterbi algorithm
Two tasks

- Parameter learning from partial observations as prefixes
  - $\sum_u P_{\text{prefix}}(u) > 1$, so prefixes cannot have a distribution and no MLE possible
  - But parameter learning by maximizing $\prod_u P_{\text{prefix}}(u | \theta)$ is still possible using the EM-like algorithm \(\Rightarrow\) Prefix-EM algorithm
  - Viterbi training from prefixes is possible \(\Rightarrow\) Prefix-VT algorithm derived
  - We compare three learning methods (Prefix-EM, Prefix-VT, PCFG-EM) by classifying prefixes of NASA dataset into five goals (survey, news, ...)

- Online goal and plan recognition from partial obs. as prefixes
  - To classify prefixes of Web log data into five goals (survey, news, ...) and infer their most likely partial plan
  - We compare four recognition methods (Prefix, HMM, LR, SVM)
  - An example of online goal recognition
The Prefix EM algorithm

**Step 1:** For each \( t (1 \leq t \leq T) \), compute an explanation graph \( \text{exp}(G_t) \) and topologically sort the SCCs extracted from \( \text{exp}(G_t) \) in descending order in terms of the \( > \) relation. Let \( \text{SCC}^1_t > \cdots > \text{SCC}^{k^*}_t \) be the sorted SCCs. \( \text{SCC}^1_t \) contains the top-goal \( G_t \) such that \( P(G_t) = P(1) \).

Repeat Step 2 until \( L = \prod_{t=1}^T P(G_t) \) converges.

**Step 2-1 (computing inside probabilities):**

For each \( t (1 \leq t \leq T) \) and for each \( k \) from \( N_t \) to \( k = 1 \), solve a set of probability equations \( X = MX + Y \) for \( \text{SCC}^k_t \) and obtain the inside probabilities \( X \) of goals in \( \text{SCC}^k_t \). Store the matrix \( M \) for later use at Step 2-2.

**Step 2-2 (computing outside probabilities):**

Put \( E[\sigma(i_d, v) \mid G_t] = 0 \) for every \( \text{msw}(i_d, v) \) and every \( t (1 \leq t \leq T) \).

For each \( t (1 \leq t \leq T) \), do the following.

For every \( j \), \( P(1)_j = 0 \). Goals in \( \text{exp}(G_t) \) are numbered so that 1 = \( G_t \). \( P(i) \) is the prob. of goal \( i \) and \( P(i)_j = \frac{\partial P(i)}{\partial P(j)} \).

For every goal \( P(n) \) in \( \text{SCC}^1_t \), set \( P(n)_n = 1 \).

For each \( k \) from 1 to \( N_t \), do the following.

Let \( X = MX + Y \) be a set of probability equations for \( \text{SCC}^k_t \).

For every \( P(j) \) appearing in \( M \) or \( Y \), solve

\[
\frac{\partial X}{\partial P(j)} = M \frac{\partial X}{\partial P(j)} + \frac{\partial M}{\partial P(j)} X + \frac{\partial Y}{\partial P(j)}
\]

and store all the \( P(i)_j \)'s in \( \frac{\partial X}{\partial P(j)} \).

For every \( P(i) \) in \( X \), increment the outside probability \( P(1)_j \) by

\[
P(1)_j \leftarrow P(1)_j \cdot P(i)_j + P(1)_j,
\]

For each \( m = \text{msw}(i_d, v) \) in \( M \) or \( Y \), increment \( E[\sigma(i_d, v) \mid G_t] \) by

\[
E[\sigma(i_d, v) \mid G_t] \leftarrow \frac{P(m) \cdot P(1)_n}{P(G_t)} + E[\sigma(i_d, v) \mid G_t]
\]

**Step 2-3:** For each \( \text{msw}(i_d, v) \) in a program, update \( \theta_{i_d, v} = P(\text{msw}(i_d, v)) \) by

\[
\theta_{i_d, v} = \frac{\sum_{t=1}^T E[\sigma(i_d, v) \mid G_t] + \alpha_{i_d, v}}{\sum_v \sum_{t=1}^T E[\sigma(i_d, v) \mid G_t] + \alpha_{i_d, v}}
\]

where \( \alpha_{i_d, v} \) is a pseudo count for \( \text{msw}(i_d, v) \).
Parameter learning (NASA)

Accuracy versus Prefix length for different methods:
- PCFG-EM
- Prefix-EM
- Prefix-VT
- Random

The graph shows the accuracy for each method across different prefix lengths, ranging from 2 to 20. The accuracy is measured in a 5-fold cross-validation (CV) setting.
Online goal recognition

Infer the visitor’s intention (survey, news, survey-specific-areas, news-specific-areas, others)

U of S

ClarkNet

NASA

Prefix length

Accuracy

5.12 \times 10^4

2.77 \times 10^5

3.14 \times 10^6

(PCFG’s entropy = - \sum_t p(t) \log p(t) of PCFG[Chi+99])

ILP 2015, Kyoto
Online plan recognition

- Prefix ➔ current goal
Conclusion

- Emerging technologies
  - will combine big data, propositions, logic, probability and KB
  - need of a solid framework unifying them

- Distribution semantics
  - computational adaptation of Fenstad’s representation theorem to LP
  - framework unifying logic and probability for probabilistic ILP
  - semantic basis for PRISM, ProbLog, LPADs, PITA, ...

- PRISM
  - an implementation of the distribution semantics
  - aims to provide one-stop service for logic-based probabilistic modeling
  - PRISM2.2 beta can compute and learn cyclic relational models that have been relatively unexplored