Declarative Machine Learning and Data Mining

Luc De Raedt
CP for Analytics Workshop, Cork, 2015
Joint work with Angelika Kimmig, Siegfried Nijssen, Tias Guns and many others
Motivation

• Use of declarative methods, a new trend in machine learning and data mining

• Two sources of inspiration
  • Stephen Boyd on Convex Optimisation for Statistical Learning
  • Helmut Simonis in CP

• Introduce two types of declarative languages for ML & DM, logic / symbolic oriented rather than pure linear algebra
  • CP4IM & MiningZinc — CP for pattern mining
  • Probabilistic Programming — in ProbLog
Declarative Mining
Patterns

Mutagenic

Clean

Clean

Mutagenic

Mutagenic
Pattern Mining

A. frequent pattern

- which patterns are frequent?

\[ Th(\mathcal{L}, Q, D) = \{ p \in \mathcal{L} | Q(p, D) = true \} \]

B. Correlated pattern mining = subgroup discovery

- which patterns are significant w.r.t. classes? all patterns? k-best patterns?

\[ Th(\mathcal{L}, Q, D) = \text{arg}\max_{p \in \mathcal{L}} \phi(p, D) \]

C. pattern set mining

- which pattern set is the best concept-description for the actives? for the inactives?

\[ Th(\mathcal{L}, Q, D) = \{ P \subseteq \mathcal{L} | Q(P, D) = true \} \]
Itemset mining

Data

Frequent patterns

4
2
3
A. Frequent Itemset Mining

Given

- \( \mathcal{I} = \{1, \cdots, NrI\} \)
  set of items

- \( \mathcal{T} = \{1, \cdots, NrT\} \)
  set of transactions identifiers

- \( \mathcal{D} = \{(t, I)|t \in \mathcal{T}, I \subseteq \mathcal{I}\} \)
  Dataset

- \( Items \subseteq \mathcal{I} \) and \( Trans \subseteq \mathcal{T} \)

Find \( Items \) such that
\[
|\text{covers}(Items, \mathcal{D})| > freq
\]

where \( \text{covers}(Items, \mathcal{D}) = \{t \in \mathcal{T}|(t, I) \in \mathcal{D} \text{ and } Items \subseteq I\} \)
A. Frequent Itemset Mining

Given

- \( \mathcal{I} = \{1, \cdots, NrI\} \)
  set of items

- \( \mathcal{I} = \{1, \cdots, NrI\} \)
  set of transactions identifiers

- \( \mathcal{D} = \{(t, I) | t \in \mathcal{T}, I \subseteq \mathcal{I}\} \)
  Dataset

- \( \text{Items} \subseteq \mathcal{I} \) and \( \text{Trans} \subseteq \mathcal{T} \)

Find \( \text{Items} \) such that

\[ |\text{covers}(\text{Items}, \mathcal{D})| > \text{freq} \]

where \( \text{covers}(\text{Items}, \mathcal{D}) = \{t \in \mathcal{T} | (t, I) \in \mathcal{D} \text{ and } \text{Items} \subseteq I\} \)

int: Freq;
int: NrI;
int: NrT;

array[1..NrT] of set of 1..NrI: D;

var set of 1..NrI: Items;
var set of 1..NrT: Trans;

constraint card(Trans) > Freq;
constraint Trans = covers(Items, D);
solve satisfy;

function var set of int: cover(Items, D) =
let {
  var set of int: Trans,
  constraint forall (t in ub(Trans))
  (t in Trans \leftrightarrow \text{Items subset D}[t])
} in Trans;
Frequent Itemset Mining

math like notation

user defined functions and constraints

solver independent (standardized)

efficiently solvable

int: Freq;
int: NrI;
int: NrT;

array[1..NrT] of set of 1..NrI: D;

var set of 1..NrI: Items;
var set of 1..NrT: Trans;

constraint card(Trans) > Freq;
constraint Trans = covers(Items, D);
solve satisfy;

function var set of int: cover(Items, D) =
let {
  var set of int: Trans,
  constraint forall (t in ub(Trans))
  (t in Trans ↔ Items subset D[t])
} in Trans;
Encoding in Zinc

int: Freq;
int: NrI; int: NrT;
array [1..NrT] of set of int: D;

array [1..NrI] of var bool: Items;
array [1..NrT] of var bool: Trans;

constraint % encode D: every Trans complement has no supported Items
   forall(t in 1..NrT) ( 
      Trans[t] <-> sum(i in 1..NrI) ( Items[i]*(1 - (i in D[t])) ) <= 0 
   );

constraint % frequency: every Item is supported by sufficiently many Trans
   forall(i in 1..NrI) ( 
      Items[i] -> sum(t in 1..NrT) ( Trans[t]*(i in D[t]) ) >= Freq 
   );

solve satisfy;

∀t : T_t = 1 ⇔ ∑_i I_i (1 − D_{ti}) = 0

∑_t T_t ≥ \text{minsup} \quad \text{iff} \quad \forall i : I_i = 1 ⇒ ∑_t T_t D_{ti} ≥ \text{minsup}
Resulting Search Strategy akin to Zaki’s Eclat [KDD 97]
Closed Itemset Mining

```plaintext
int: Freq;
int: Nrl;
int: NrT;

array[1..NrT] of set of 1..Nrl: D;

var set of 1..Nrl: Items;
var set of 1..NrT: Trans;

constraint card(Trans) > Freq;
constraint Trans = covers(Items, D);
constraint Items = cover_inv(Trans, D);
solve satisfy;

function var set of int: cover_inv(Trans, D) =
let {
    var set of int: Items,
    constraint forall (i in ub(Items))
        (i in Items ↔ Trans subset D'[i] )
} in Items;

function var set of int: cover(Items, D) =
let {
    var set of int: Trans,
    constraint forall (t in ub(Trans))
        (t in Trans ↔ Items subset D[t] )
} in Trans;
```
Further Constraints

* exact coverage:
  \[ t \in \text{Trans} \iff \text{Items} \subseteq D[t] \]

* freq:
  \[ i \in \text{Items} \implies \text{card}(\text{Trans} \cap D'[i]) \geq Freq \]

* maximal:
  \[ i \in \text{Items} \iff \text{card}(\text{Trans} \cap D'[i]) \geq Freq \]

* closed:
  \[ i \in \text{Items} \iff \text{Trans} \subseteq D'[i] \]

* delta-closed:
  \[ i \in \text{Items} \iff \text{card}(\text{Trans} \cap D'[i]) \leq \Delta \cdot \text{card}(\text{Trans}) \]
Top-k Correlated Pattern Mining

- $D$ now consists of two datasets, say $P$ and $N$
- a correlation function $\phi(p, D)$, e.g., $\chi^2$
- $Th(\mathcal{L}, Q, D) = \arg_{p \in \mathcal{L}} \max_k \phi(p, D)$
Modeling perspective

Alternative opt. functions, for example:

```
solve maximize chi2(Trans, pos, neg);
```

with:
```
function float: chi2(Trans, pos, neg)
```
Correlation function

Figure 1: A plot of the $\chi^2$ scoring function, and a threshold on $\chi^2$. 
## Generality

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum frequency</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Maximum frequency</td>
<td></td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Emerging patterns</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Condensed Representations</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Maximal</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Closed</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(\delta)-Closed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Constraints on syntax</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Max/Min total cost</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Minimum average cost</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Max/Min size</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Constraints on labelled data</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Minimum correlation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Maximum correlation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

[L. De Raedt, T. Guns, S. Nijssen, AAAI 2010]
Declarative constraint-based mining

Language goals:
- high-level notation (similar to paper definitions)
- solver-independent
- user-defined abstractions
- mathematical-like language (Zinc)
- many solvers
- can define custom predicates \textit{and} functions


$\Rightarrow$ MiningZinc

1) Normalize to FlatZinc *(do not flatten lib_itemsetmining.mzn yet)*

2) Apply rewrite rules to:
   - add redundant constraints
   - detect (partial) applicability of specialised algorithms
   - tailor to constraint solvers

3) Collect all feasible rewrite combinations = execution plans

4) Heuristically rank + execute a plan
Three categories of execution plans:

A) **Specialised algorithms only**
   - Eclat-maxfreq(TDB, 20, 50)
   - LCMv2(TDB, 20) + maxcover(Items, TDB, 40)

B) **Hybrid decomposition**
   - LCMv2(TDB, 20) + gecode(card(cover(Items, TDB)) =< 40)
   - LCMv2(TDB, 20) + frequency(Items, TDB, S) + gecode(S =< 40)

C) **Generic solvers only**
   - gecode(…)
   - gecode-bool(…)
   - gecode-bool(… + redundant)
   - or-tools-bool(…)

```plaintext
var set of 1..Nrl: Items; array[int] of set of int: TDB;
constraint card(cover(Items, TDB)) >= 20;
constraint card(cover(Items, TDB)) =< 40;
solve satisfy;
```
Experiments, hybrid solving

frequent itemset mining, with minimum size and closure constraint
Software

MiningZinc, this high-level declarative framework for constraint-based mining is available on a separate page.

Download FIM_CP

The frequent and constraint-based itemset mining system using Constraint Programming.

- Latest version: fimcp-2.7.tar.gz
- License: MIT license
- Language: C++ (tested on Linux, known to work under Mac and Windows)
- Latest changes: Release with new, easier, build system: it now automatically downloads, configures and compiles the Gecode CP solver (version 3.7.1).

Installation instructions in the accompanying README. See online usage instructions and examples.

Basic usage

MiningZinc can be used through two interfaces: as a command line tool and as a Python package.

From the command line

MiningZinc can be run through its command line interface. There are four modes:

- list: analyze the model and list the possible execution plans
- solve: solve the model
- interactive: combination of previous two modes with a choice menu
- default: solve the model using the first available strategy

The default usage of MiningZinc is:

```
./miningzinc model.mzn data1.dzn data2.dzn -D "Param1=10;"
```
State of the art

• Many other works
  • Clustering (Ian Davidson, Christel Vrain, Thi-Bich-Hanh Dao, …)
  • Skyline patterns (B. Cremilleux, … ; Negrevergne, …)
  • Other pattern types (L. Sais, ..; Guns, …)

• Challenges
  • Scaling
  • Structured data (graphs, sequences, …)
  • Integration with some other ML/DM techniques
  • Other types of solvers …
Probabilistic Programming
World Dynamics

Fragment of world with

~10 alliances
~200 players
~600 cities

alliances color-coded

Can we build a model
of this world?
Can we use it for playing
better?

[Thon, Landwehr, De Raedt, ECML08]
World Dynamics

Fragment of world with

~10 alliances
~200 players
~600 cities

alliances color-coded

Can we build a model of this world?
Can we use it for playing better?

[Thon, Landwehr, De Raedt, ECML08]
World Dynamics

Fragment of world with

~10 alliances
~200 players
~600 cities

alliances color-coded

Can we build a model of this world? Can we use it for playing better?

[Thon, Landwehr, De Raedt, ECML08]
### Recently-Learned Facts

<table>
<thead>
<tr>
<th>instance</th>
<th>iteration</th>
<th>date learned</th>
<th>confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>kelly_andrews is a female</td>
<td>826</td>
<td>29-mar-2014</td>
<td>98.7</td>
</tr>
<tr>
<td>investment_next_year is an economic sector</td>
<td>829</td>
<td>10-apr-2014</td>
<td>95.3</td>
</tr>
<tr>
<td>shibenik is a geopolitical entity that is an organization</td>
<td>829</td>
<td>10-apr-2014</td>
<td>97.2</td>
</tr>
<tr>
<td>quality_web_design_work is a character trait</td>
<td>826</td>
<td>29-mar-2014</td>
<td>91.0</td>
</tr>
<tr>
<td>mercedes_benz_cls_by_carlsson is an automobile manufacturer</td>
<td>829</td>
<td>10-apr-2014</td>
<td>95.2</td>
</tr>
<tr>
<td>social_work is an academic program at the university rutgers_university</td>
<td>827</td>
<td>02-apr-2014</td>
<td>93.8</td>
</tr>
<tr>
<td>dante wrote the book the_divine_comedy</td>
<td>826</td>
<td>29-mar-2014</td>
<td>93.8</td>
</tr>
<tr>
<td>willie_aames was born in the city los_angeles</td>
<td>831</td>
<td>16-apr-2014</td>
<td>100.0</td>
</tr>
<tr>
<td>kitt_peak is a mountain in the state or province arizona</td>
<td>831</td>
<td>16-apr-2014</td>
<td>96.9</td>
</tr>
<tr>
<td>greenwich is a park in the city london</td>
<td>831</td>
<td>16-apr-2014</td>
<td>100.0</td>
</tr>
</tbody>
</table>

NELL: http://rtw.ml.cmu.edu/rtw/
Example: Information Extraction

<table>
<thead>
<tr>
<th>instance</th>
<th>iteration</th>
<th>date learned</th>
<th>confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>kelly_andrews is a female</td>
<td>826</td>
<td>29-mar-2014</td>
<td>98.7</td>
</tr>
<tr>
<td>investment_next_year is an economic sector</td>
<td>829</td>
<td>10-apr-2014</td>
<td>95.3</td>
</tr>
<tr>
<td>shibenik is a geopolitical entity that is an organization</td>
<td>829</td>
<td>10-apr-2014</td>
<td>97.2</td>
</tr>
<tr>
<td>quality_web_design_work is a character trait</td>
<td>826</td>
<td>29-mar-2014</td>
<td>91.0</td>
</tr>
<tr>
<td>mercedes_benz cls by carlsson is an automobile manufacturer</td>
<td>829</td>
<td>10-apr-2014</td>
<td>95.2</td>
</tr>
<tr>
<td>social_work is an academic program at the university rutgers university</td>
<td>827</td>
<td>02-apr-2014</td>
<td>93.8</td>
</tr>
<tr>
<td>dante wrote the book the divine comedy</td>
<td>826</td>
<td>29-mar-2014</td>
<td>93.8</td>
</tr>
<tr>
<td>willie_aames was born in the city los angeles</td>
<td>831</td>
<td>16-apr-2014</td>
<td>100.0</td>
</tr>
<tr>
<td>kitt_peak is a mountain in the state or province arizona</td>
<td>831</td>
<td>16-apr-2014</td>
<td>96.9</td>
</tr>
<tr>
<td>greenwich is a park in the city london</td>
<td>831</td>
<td>16-apr-2014</td>
<td>100.0</td>
</tr>
</tbody>
</table>
**Example: Information Extraction**

instances for many different relations

<table>
<thead>
<tr>
<th>instance</th>
<th>iteration</th>
<th>date learned</th>
<th>confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>kelly_andrews is a female</td>
<td>826</td>
<td>29-mar-2014</td>
<td>98.7</td>
</tr>
<tr>
<td>investment_next_year is an economic sector</td>
<td>829</td>
<td>10-apr-2014</td>
<td>95.3</td>
</tr>
<tr>
<td>shibenik is a geopolitical entity that is an organization</td>
<td>829</td>
<td>10-apr-2014</td>
<td>97.2</td>
</tr>
<tr>
<td>quality_web_design_work is a character trait</td>
<td>826</td>
<td>29-mar-2014</td>
<td>91.0</td>
</tr>
<tr>
<td>mercedes_benz_cls_by_carlsson is an automobile manufacturer</td>
<td>829</td>
<td>10-apr-2014</td>
<td>95.2</td>
</tr>
<tr>
<td>social_work is an academic program at the university rutgers university</td>
<td>827</td>
<td>02-apr-2014</td>
<td>93.8</td>
</tr>
<tr>
<td>dante wrote the book the_dive_comedy</td>
<td>826</td>
<td>29-mar-2014</td>
<td>93.8</td>
</tr>
<tr>
<td>willie_aames was born in the city los_angeles</td>
<td>831</td>
<td>16-apr-2014</td>
<td>100.0</td>
</tr>
<tr>
<td>kitt_peak is a mountain in the state or province arizona</td>
<td>831</td>
<td>16-apr-2014</td>
<td>96.9</td>
</tr>
<tr>
<td>greenwich is a park in the city london</td>
<td>831</td>
<td>16-apr-2014</td>
<td>100.0</td>
</tr>
</tbody>
</table>

NELL: http://rtw.ml.cmu.edu/rtw/
ProbLog by example:

A bit of gambling

• toss (biased) coin & draw ball from each urn

• win if (heads and a red ball) or (two balls of same color)
ProbLog by example:

A bit of gambling

- toss (biased) coin & draw ball from each urn
- win if (heads and a red ball) or (two balls of same color)

probabilistic fact: heads is true with probability 0.4 (and false with 0.6)
ProbLog by example:

A bit of gambling

- toss (biased) coin & draw ball from each urn
- win if (heads and a red ball) or (two balls of same color)

0.4 :: heads.

annotated disjunction: first ball is red with probability 0.3 and blue with 0.7

0.3 :: col(1,red); 0.7 :: col(1,blue).
ProbLog by example:

A bit of gambling

- toss (biased) coin & **draw ball from each urn**
- win if (heads and a red ball) or (two balls of same color)

0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue).
0.2 :: col(2,red); 0.3 :: col(2,green);
   0.5 :: col(2,blue).

**annotated disjunction**: second ball is red with probability 0.2, green with 0.3, and blue with 0.5
ProbLog by example:

A bit of gambling

• toss (biased) coin & draw ball from each urn

• **win if (heads and a red ball) or (two balls of same color)**

0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue).
0.2 :: col(2,red); 0.3 :: col(2,green);
    0.5 :: col(2,blue).

**win :- heads, col(\_,red).**  **logical rule** encoding background knowledge
ProbLog by example:

A bit of gambling

- toss (biased) coin & draw ball from each urn
- \textbf{win if (heads and a red ball) or (two balls of same color)}

\begin{align*}
0.4 & :: \text{heads} . \\
0.3 & :: \text{col(1,red)} ; 0.7 & :: \text{col(1,blue)} . \\
0.2 & :: \text{col(2,red)} ; 0.3 & :: \text{col(2,green)} ; \\
& \quad 0.5 :: \text{col(2,blue)} .
\end{align*}

\textit{logical rule} encoding \\
\textit{background knowledge}
ProbLog by example:

A bit of gambling

- toss (biased) coin & draw ball from each urn
- win if (heads and a red ball) or (two balls of same color)

0.4 :: heads.

probabilistic choices

0.3 :: col(1,red); 0.7 :: col(1,blue).
0.2 :: col(2,red); 0.3 :: col(2,green);
0.5 :: col(2,blue).

win :- heads, col(_,red).
win :- col(1,C), col(2,C).
Questions

0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue).
0.2 :: col(2,red); 0.3 :: col(2,green); 0.5 :: col(2,blue).

win :- heads, col(_,red).
win :- col(1,C), col(2,C).

• Probability of win?

• Probability of win given col(2,green)?

• Most probable world where win is true?
Questions

0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue).
0.2 :: col(2,red); 0.3 :: col(2,green); 0.5 :: col(2,blue).

win :- heads, col(_,red).
win :- col(1,C), col(2,C).

marginal probability

• Probability of \texttt{win} query

• Probability of $\texttt{win}$ given $\texttt{col(2,green)}$?

• Most probable world where $\texttt{win}$ is true?
Questions

0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue).
0.2 :: col(2,red); 0.3 :: col(2,green); 0.5 :: col(2,blue).

\[ \text{win} :: \text{heads, col(\_, red)}. \]
\[ \text{win} :: \text{col(1,C), col(2,C)}. \]

- Probability of \text{win}?
- Probability of \text{win} given \text{col(2,green)}?
- Most probable world where \text{win} is true?
Questions

0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue).
0.2 :: col(2,red); 0.3 :: col(2,green); 0.5 :: col(2,blue).

win :- heads, col(_,red).
win :- col(1,C), col(2,C).

marginal probability

- Probability of win?

conditional probability

- Probability of win given col(2,green)?

- Most probable world where win is true?

MPE inference
Possible Worlds

0.4 :: heads.

0.3 :: col(1, red); 0.7 :: col(1, blue).
0.2 :: col(2, red); 0.3 :: col(2, green); 0.5 :: col(2, blue).

win :- heads, col(_, red).
win :- col(1, C), col(2, C).
Possible Worlds

0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue).
0.2 :: col(2,red); 0.3 :: col(2,green); 0.5 :: col(2,blue).

win :- heads, col(_,red).
win :- col(1,C), col(2,C).
Possible Worlds

0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue).
0.2 :: col(2,red); 0.3 :: col(2,green); 0.5 :: col(2,blue).

win :- heads, col(_,red).
win :- col(1,C), col(2,C).
Possible Worlds

0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue).
0.2 :: col(2,red); 0.3 :: col(2,green); 0.5 :: col(2,blue).

win :- heads, col(_,red).
win :- col(1,C), col(2,C).

0.4 × 0.3

H  R
Possible Worlds

0.4 :: heads.
0.3 :: col(1,red); 0.7 :: col(1,blue).
0.2 :: col(2,red); 0.3 :: col(2,green); 0.5 :: col(2,blue).

win :- heads, col(_,red).
win :- col(1,C), col(2,C).

0.4 \times 0.3 \times 0.3
Possible Worlds

0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue).
0.2 :: col(2,red); 0.3 :: col(2,green); 0.5 :: col(2,blue).

\[\text{win} :- \text{heads}, \text{col}(_,\text{red}).\]
\[\text{win} :- \text{col}(1,C), \text{col}(2,C).\]

\[0.4 \times 0.3 \times 0.3\]
Possible Worlds

0.4 :: heads.
0.3 :: col(1,red); 0.7 :: col(1,blue).
0.2 :: col(2,red); 0.3 :: col(2,green); 0.5 :: col(2,blue).

win :- heads, col(_,red).
win :- col(1,C), col(2,C).
0.4 :: heads.

0.3 :: col(1, red); 0.7 :: col(1, blue).
0.2 :: col(2, red); 0.3 :: col(2, green); 0.5 :: col(2, blue).

\[ \text{win} \leftarrow \text{heads}, \text{col}(\_, \text{red}). \]
\[ \text{win} \leftarrow \text{col}(1, C), \text{col}(2, C). \]
Possible Worlds

0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue).
0.2 :: col(2,red); 0.3 :: col(2,green); 0.5 :: col(2,blue).

win :- heads, col(_,red).
win :- col(1,C), col(2,C).
Possible Worlds

0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue).
0.2 :: col(2,red); 0.3 :: col(2,green); 0.5 :: col(2,blue).

\[
\text{win} :- \text{heads}, \text{col}(_,\text{red}).
\]
\[
\text{win} :- \text{col}(1,C), \text{col}(2,C).
\]

\[
0.4 \times 0.3 \times 0.3 \quad (1-0.4) \times 0.3 \times 0.2
\]

\[
\begin{array}{c}
\text{H} \\
\text{R} \\
\text{G} \\
\text{W}
\end{array} 
\quad  
\begin{array}{c}
\text{R} \\
\text{R} \\
\text{W}
\end{array}
\]
Possible Worlds

0.4 :: heads.

0.3 :: col(1, red); 0.7 :: col(1, blue).
0.2 :: col(2, red); 0.3 :: col(2, green); 0.5 :: col(2, blue).

\[ \text{win} :: \text{heads}, \text{col}(_, \text{red}). \]
\[ \text{win} :: \text{col}(1, \text{C}), \text{col}(2, \text{C}). \]

\[ 0.4 \times 0.3 \times 0.3 \quad (1 - 0.4) \times 0.3 \times 0.2 \quad (1 - 0.4) \]
Possible Worlds

0.4 :: heads.

0.3 :: col(1, red); 0.7 :: col(1, blue).
0.2 :: col(2, red); 0.3 :: col(2, green); 0.5 :: col(2, blue).

\[
\text{win} :- \text{heads, col(\_, red)}.
\]
\[
\text{win} :- \text{col}(1, C), \text{col}(2, C).
\]
Possible Worlds

0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue).
0.2 :: col(2,red); 0.3 :: col(2,green); 0.5 :: col(2,blue).

win :- heads, col(_,red).
win :- col(1,C), col(2,C).

\[
\begin{align*}
0.4 \times 0.3 \times 0.3 & \quad (1-0.4) \times 0.3 \times 0.2 \\
0.4 \times 0.3 \times 0.3 & \quad (1-0.4) \times 0.3 \times 0.3
\end{align*}
\]
Possible Worlds

0.4 :: heads.

0.3 :: col(1, red); 0.7 :: col(1, blue).
0.2 :: col(2, red); 0.3 :: col(2, green); 0.5 :: col(2, blue).

\[
\begin{align*}
\text{win} & :- \text{heads, col(\_\_, red)}.
\text{win} & :- \text{col(1,C), col(2,C)}.
\end{align*}
\]

\[
\begin{align*}
0.4 \times 0.3 \times 0.3 & \quad (1-0.4) \times 0.3 \times 0.2 & \quad (1-0.4) \times 0.3 \times 0.3
\end{align*}
\]
Possible Worlds

0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue).
0.2 :: col(2,red); 0.3 :: col(2,green); 0.5 :: col(2,blue).

win :- heads, col(_,red).
win :- col(1,C), col(2,C).
All Possible Worlds

0.024

0.036

0.056

0.084

0.036

0.054

0.084

0.126

0.060

0.090

0.140

0.210
Most likely world where win is true?

<table>
<thead>
<tr>
<th>Probability</th>
<th>World 1</th>
<th>World 2</th>
<th>World 3</th>
<th>World 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.024</td>
<td>H R R</td>
<td>H R R</td>
<td>H B R</td>
<td>B R</td>
</tr>
<tr>
<td>0.036</td>
<td>H R G</td>
<td>R G</td>
<td>H B G</td>
<td>B G</td>
</tr>
<tr>
<td>0.060</td>
<td>H R B</td>
<td>R B</td>
<td>H B B</td>
<td>B B</td>
</tr>
</tbody>
</table>
Most likely world where win is true?

MPE Inference
Most likely world where \texttt{col(2,blue)} is false?

\begin{align*}
0.024 & : \begin{array}{c}
H \quad R \\ W
\end{array} & 0.036 & : \begin{array}{c}
R \quad R \\ W
\end{array} & 0.056 & : \begin{array}{c}
H \quad B \\ W
\end{array} & 0.084 & : \begin{array}{c}
B \quad R \\ W
\end{array} \\
0.036 & : \begin{array}{c}
H \quad R \\ W
\end{array} & 0.054 & : \begin{array}{c}
R \quad G \\ W
\end{array} & 0.084 & : \begin{array}{c}
H \quad B \\ W
\end{array} & 0.126 & : \begin{array}{c}
B \quad G \\ W
\end{array} \\
0.060 & : \begin{array}{c}
H \quad R \\ W
\end{array} & 0.090 & : \begin{array}{c}
R \quad B \\ W
\end{array} & 0.140 & : \begin{array}{c}
H \quad B \\ W
\end{array} & 0.210 & : \begin{array}{c}
B \quad B \\ W
\end{array}
\end{align*}
Most likely world where \( \text{col}(2, \text{blue}) \) is false?

- 0.024 (H R R W)
- 0.036 (H R R W)
- 0.056 (H B R W)
- 0.084 (H B R W)
- 0.036 (H R G W)
- 0.054 (H R G W)
- 0.084 (H B G W)
- 0.126 (H B G W)
- 0.060 (H R B W)
- 0.090 (H R B W)
- 0.140 (H B B W)
- 0.210 (H B B W)
\[ P(\text{win}) = ? \]

Marginal Probability

<table>
<thead>
<tr>
<th></th>
<th>0.024</th>
<th>0.036</th>
<th>0.056</th>
<th>0.084</th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>R</td>
<td>R</td>
<td>B</td>
<td>B</td>
</tr>
<tr>
<td>W</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>0.036</th>
<th>0.054</th>
<th>0.084</th>
<th>0.126</th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>R</td>
<td>R</td>
<td>B</td>
<td>B</td>
</tr>
<tr>
<td>W</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>0.060</th>
<th>0.090</th>
<th>0.140</th>
<th>0.210</th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>R</td>
<td>R</td>
<td>B</td>
<td>B</td>
</tr>
<tr>
<td>W</td>
<td>B</td>
<td>B</td>
<td>B</td>
<td>B</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
</tr>
</tbody>
</table>
\[ P(\text{win}) = \sum \]

Marginal Probability

0.024
\[
\begin{array}{c}
\text{H} \\
\text{R} \\
\text{R} \\
\text{W}
\end{array}
\]
0.036
\[
\begin{array}{c}
\text{R} \\
\text{R} \\
\text{R} \\
\text{W}
\end{array}
\]
0.056
\[
\begin{array}{c}
\text{H} \\
\text{B} \\
\text{R} \\
\text{W}
\end{array}
\]
0.084
\[
\begin{array}{c}
\text{B} \\
\text{R}
\end{array}
\]

0.036
\[
\begin{array}{c}
\text{H} \\
\text{R} \\
\text{G} \\
\text{W}
\end{array}
\]
0.054
\[
\begin{array}{c}
\text{R} \\
\text{G}
\end{array}
\]
0.084
\[
\begin{array}{c}
\text{H} \\
\text{B} \\
\text{G}
\end{array}
\]
0.126
\[
\begin{array}{c}
\text{B} \\
\text{G}
\end{array}
\]

0.060
\[
\begin{array}{c}
\text{H} \\
\text{R} \\
\text{B} \\
\text{W}
\end{array}
\]
0.090
\[
\begin{array}{c}
\text{R} \\
\text{B}
\end{array}
\]
0.140
\[
\begin{array}{c}
\text{H} \\
\text{B} \\
\text{B} \\
\text{W}
\end{array}
\]
0.210
\[
\begin{array}{c}
\text{B} \\
\text{B}
\end{array}
\]
\( P(\text{win}) = \sum = 0.562 \)
Conditional Probability

P(win|col(2, green)) = ?

<table>
<thead>
<tr>
<th>Probability</th>
<th>Color Configuration</th>
<th>Probability</th>
<th>Color Configuration</th>
<th>Probability</th>
<th>Color Configuration</th>
<th>Probability</th>
<th>Color Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.024</td>
<td>HRR</td>
<td>0.036</td>
<td>RRR</td>
<td>0.056</td>
<td>HRB</td>
<td>0.084</td>
<td>BRR</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td></td>
<td>W</td>
<td></td>
<td>W</td>
<td></td>
<td>W</td>
</tr>
<tr>
<td>0.036</td>
<td>HRG</td>
<td>0.054</td>
<td>RRG</td>
<td>0.084</td>
<td>HBG</td>
<td>0.126</td>
<td>BRG</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td></td>
<td>W</td>
<td></td>
<td>W</td>
<td></td>
<td>W</td>
</tr>
<tr>
<td>0.060</td>
<td>HRB</td>
<td>0.090</td>
<td>RRB</td>
<td>0.140</td>
<td>HBB</td>
<td>0.210</td>
<td>BBB</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td></td>
<td>W</td>
<td></td>
<td>W</td>
<td></td>
<td>W</td>
</tr>
</tbody>
</table>
\[ P(\text{win}|\text{col}(2,\text{green})) = \frac{\sum}{\sum} = \frac{P(\text{win} \land \text{col}(2,\text{green}))}{P(\text{col}(2,\text{green}))} \]
\[ P(\text{win}|\text{col}(2,\text{green})) = \frac{\sum}{\sum} = \frac{P(\text{win} \land \text{col}(2,\text{green}))}{P(\text{col}(2,\text{green}))} \]
\[ P(\text{win}|\text{col}(2,\text{green})) = \frac{\sum}{\sum} = 0.036/0.3 = 0.12 \]
Inference in PLP

• As in Prolog and logic programming
  • proof-based
• As in Answer Set Programming
  • model based
• As in Probabilistic Programming
  • sampling
Parameter Learning

e.g., webpage classification model

for each \textit{CLASS1, CLASS2} and each \textit{WORD}

\begin{itemize}
  \item \texttt{link\_class(Source,Target,CLASS1,CLASS2)}.
  \item \texttt{word\_class(WORD,CLASS)}.
\end{itemize}

\texttt{class(Page,C) :- has\_word(Page,W), word\_class(W,C).}

\texttt{class(Page,C) :- links\_to(OtherPage,Page),}
\texttt{ class(OtherPage,OtherClass),
  link\_class(OtherPage,Page,OtherClass,C).}
Sampling

Interpretations
Sampling Interpretations
Parameter Estimation
Parameter Estimation

\[ p(\text{fact}) = \frac{\text{count(\text{fact is true})}}{\text{Number of interpretations}} \]
Learning from partial interpretations

- Not all facts observed
- Soft-EM
- use expected count instead of count
- \( P(Q | E) \) -- conditional queries!

[Gutmann et al, ECML 11; Fierens et al, TPLP 14]
Bayesian Parameter Learning

• Learning as inference (e.g., Church)
• Prior distributions for parameters
• Given data, find most likely parameter values
Example

- Flipping a coin with unknown weight
- Prior: uniform distribution on \([0,1]\)
- Observation: 5x heads in a row
- Sampling from Church model:
ProbLog Example

Prior

0.05::weight(C,0.1); 0.2::weight(C,0.3); 0.5::weight(C,0.5);
0.2::weight(C,0.7); 0.05::weight(C,0.9) :- coin(C).

Param::toss(_,Param,_,_).
heads(C,R) :- weight(C,Param),toss(C,Param,R).
tails(C,R) :- weight(C,Param),\+toss(C,Param,R).

data(C,[]).
data(C,[h|R]) :- heads(C,R), data(C,R).
data(C,[t|R]) :- tails(C,R), data(C,R).

Ask for posterior

query(weight(C,X)) :- coin(C),param(X).
evidence(data(c1,[h,h,h,h,h,h,h,h,h,h,h,h,h]),true).
evidence(data(c2,[h,t,h,h,h,h,t,t,h,t,t,h]),true).

data
ProbLog Example

query(weight(C,X)) :- coin(C),param(X).

evidence(data(c1,[h,h,h,h,h,h,h,h,h,h,h,h,h]),true).
evidence(data(c2,[h,t,h,h,h,h,t,t,h,t,t,h]),true).

prior

posterior for c1

posterior for c2
Some Probabilistic Programming Languages outside LP

- IBAL [Pfeffer 01]
- Figaro [Pfeffer 09]
- Church [Goodman et al 08]
- BLOG [Milch et al 05]
- Venture [Mansingha et al.]
- Anglican and Probabilistic-C [Wood et al].
- and many more appearing recently
Church by example:

A bit of gambling

- toss (biased) coin & draw ball from each urn
- win if (heads and a red ball) or (two balls of same color)
Church by example:

A bit of gambling

- toss (biased) coin & draw ball from each urn
- win if (heads and a red ball) or (two balls of same color)
Church by example:

A bit of gambling

• toss (biased) coin & draw ball from each urn

• win if (heads and a red ball) or (two balls of same color)

(define heads (mem (lambda () (flip 0.4)))))
Church by example:

A bit of gambling

- toss (biased) coin & draw ball from each urn
- win if (heads and a red ball) or (two balls of same color)

(define heads (mem (lambda () (flip 0.4))))
Church by example:

A bit of gambling

- toss (biased) coin & **draw ball from each urn**
- win if (heads and a red ball) or (two balls of same color)

```
(define heads (mem (lambda () (flip 0.4))))
(define color1 (mem (lambda () (if (flip 0.3) 'red 'blue))))
```
Church by example:

A bit of gambling

- toss (biased) coin & draw ball from each urn
- win if (heads and a red ball) or (two balls of same color)

```
(define heads (mem (lambda () (flip 0.4)))))
(define color1 (mem (lambda () (if (flip 0.3) 'red 'blue)))))
(define color2 (mem (lambda ()
    (multinomial '(red green blue) '(0.2 0.3 0.5)))))
```
Church by example:

A bit of gambling

• toss (biased) coin & draw ball from each urn

• **win if (heads and a red ball) or (two balls of same color)**

```
(define heads (mem (lambda () (flip 0.4))))
(define color1 (mem (lambda () (if (flip 0.3) 'red 'blue))))
(define color2 (mem (lambda ()
            (multinomial '(red green blue) '(0.2 0.3 0.5)))))
```
Church by example:

A bit of gambling

- toss (biased) coin & draw ball from each urn
- **win if (heads and a red ball) or (two balls of same color)**

```
(define heads (mem (lambda () (flip 0.4))))
(define color1 (mem (lambda () (if (flip 0.3) 'red 'blue))))
(define color2 (mem (lambda ()
                          (multinomial '(red green blue) '(0.2 0.3 0.5)))))
(define redball (or (equal? (color1) 'red) (equal? (color2) 'red)))
```
Church by example:

**A bit of gambling**

- toss (biased) coin & draw ball from each urn
- **win if (heads and a red ball) or (two balls of same color)**

```scheme
(define heads (mem (lambda () (flip 0.4)))))
(define color1 (mem (lambda () (if (flip 0.3) 'red 'blue)))))
(define color2 (mem (lambda ()
  (multinomial '(red green blue) '(0.2 0.3 0.5))))))
(define redball (or (equal? (color1) 'red) (equal? (color2) 'red)))
(define win1 (and (heads) redball))
```
Church by example:

A bit of gambling

• toss (biased) coin & draw ball from each urn

• **win if** (heads and a red ball) or **(two balls of same color)**

(define heads (mem (lambda () (flip 0.4))))

(define color1 (mem (lambda () (if (flip 0.3) 'red 'blue))))

(define color2 (mem (lambda ()
    (multinomial '(red green blue) '(0.2 0.3 0.5)))))

(define redball (or (equal? (color1) 'red) (equal? (color2) 'red)))

(define win1 (and (heads) redball))
Church by example:

A bit of gambling

- toss (biased) coin & draw ball from each urn
- win if (heads and a red ball) or (two balls of same color)

```scheme
(define heads (mem (lambda () (flip 0.4))))
(define color1 (mem (lambda () (if (flip 0.3) 'red 'blue))))
(define color2 (mem (lambda ()
    (multinomial '(red green blue) '(0.2 0.3 0.5))))))
(define redball (or (equal? (color1) 'red) (equal? (color2) 'red)))
(define win1 (and (heads) redball))
(define win2 (equal? (color1) (color2)))
```
Church by example:

A bit of gambling

- toss (biased) coin & draw ball from each urn
- **win if** (heads and a red ball) or (two balls of same color)

```scheme
(define heads (mem (lambda () (flip 0.4)))))
(define color1 (mem (lambda () (if (flip 0.3) 'red 'blue)))))
(define color2 (mem (lambda ()
  (multinomial '(red green blue) '(0.2 0.3 0.5)))))
(define redball (or (equal? (color1) 'red) (equal? (color2) 'red)))
(define win1 (and (heads) redball))
(define win2 (equal? (color1) (color2)))
```
Church by example:

A bit of gambling

• toss (biased) coin & draw ball from each urn

  • win if (heads and a red ball) or (two balls of same color)

(define heads (mem (lambda () (flip 0.4)))))
(define color1 (mem (lambda () (if (flip 0.3) 'red 'blue)))))
(define color2 (mem (lambda ()
  (multinomial '(red green blue) '(0.2 0.3 0.5)))))
(define redball (or (equal? (color1) 'red) (equal? (color2) 'red)))
(define win1 (and (heads) redball))
(define win2 (equal? (color1) (color2)))
(define win (or win1 win2))
Church by example:

A bit of gambling

• toss (biased) coin & draw ball from each urn

• win if (heads and a red ball) or (two balls of same color)

(define heads (mem (lambda () (flip 0.4)))))
(define color1 (mem (lambda () (if (flip 0.3) 'red 'blue)))))
(define color2 (mem (lambda ()
  (multinomial '(red green blue) '(0.2 0.3 0.5)))))
(define redball (or (equal? (color1) 'red) (equal? (color2) 'red)))
(define win1 (and (heads) redball))
(define win2 (equal? (color1) (color2)))
(define win (or win1 win2))
Probabilistic Programming Summary

• Programming language + random primitives

• probabilistic generative model

• sampling or exact inference

• learning

• increasing number of probabilistic programming languages using various underlying paradigms

• Challenge : inference & scaling up …
Introduction.

Probabilistic logic programs are logic programs in which some of the facts are annotated with probabilities.

ProbLog is a tool that allows you to intuitively build programs that do not only encode complex interactions between a large sets of heterogenous components but uncertainties that are present in real-life situations.

The engine tackles several tasks such as computing the marginals given evidence and learning from (partial) interpretations. ProbLog is a suite of efficient algorithms tasks. It is based on a conversion of the program and the queries and evidence to a weighted Boolean formula. This allows us to reduce the inference tasks to well-s weighted model counting, which can be solved using state-of-the-art methods known from the graphical model and knowledge compilation literature.

The Language. Probabilistic Logic Programming.

ProbLog makes it easy to express complex, probabilistic models.

\[
0.3::\text{stress}(X) \leftarrow \text{person}(X).
0.2::\text{influences}(X,Y) \leftarrow \text{person}(X), \text{person}(Y).
\]

\[
\text{smokes}(X) \leftarrow \text{stress}(X).
\text{smokes}(X) \leftarrow \text{friend}(X,Y), \text{influences}(Y,X), \text{smokes}(Y).
\]