A tutorial on logic-based approaches to SRL

James Cussens
University of York, UK

SIM 2009, Joint Conference SRL-ILP-MLG
Overview

- Some connections between logic and probability
- Briefly: first-order probabilistic models using ‘parfactors’ (e.g. Markov logic)
- In more detail: a generative approach for first-order probabilistic models (PRISM)
- An example of using SRL in statistical genetics
Making the connections

First-order logic \(\longrightarrow\) First-order probabilistic models

Propositional logic \(\longrightarrow\) Propositional probabilistic models
Propositional logic

First-order logic \rightarrow \text{First-order probabilistic models}

\uparrow

Propositional logic \rightarrow \text{Propositional probabilistic models}
Propositional formulae as zero-one factors

- Propositional atoms are binary (0-1) variables.
- A joint instantiation of all atoms/variables satisfying a propositional formula is a *model* of that formula.
- If $A$ and $B$ are the only propositions in our language then $A, \neg A \lor B$ has only one model.

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>A</th>
<th>B</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

James Cussens University of York, UK
Propositional probabilistic models

First-order logic $\rightarrow$ First-order probabilistic models

Propositional logic $\rightarrow$ Propositional probabilistic models
Generalising propositional logic

- Allow arbitrary non-negative values in the factors.
- Allow variables to have more than 2 values.

\[
\begin{array}{c|c|c|c|c|c|c}
A & A & B & A & B & \text{total} \\
- & - & - & - & - & - \\
0 & 4 & * & 0 & 0 & 1 & 5 &= 0 & 0 & 20 \\
1 & 6 & 0 & 1 & 5 & 0 & 1 & 20 \\
1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
1 & 1 & 7 & 1 & 1 & 42 & \\
\end{array}
\]

Dividing by a normalising constant \(Z\) defines a probability distribution over full joint instantiations (when \(Z > 0\)). Here \(Z = 20 + 20 + 0 + 42 = 82\).
### Further examples

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>A</th>
<th>D</th>
<th>B</th>
<th>C</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0.46 0</td>
<td>0</td>
<td>0.90 0</td>
<td>0</td>
<td>0</td>
<td>0.40 0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0.92 * 0</td>
<td>1</td>
<td>0.20 0</td>
<td>1</td>
<td>0</td>
<td>0.30 0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1.39 1</td>
<td>0</td>
<td>0.70 1</td>
<td>0</td>
<td>0</td>
<td>0.30 1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0.92 1</td>
<td>1</td>
<td>0.10 1</td>
<td>1</td>
<td>0</td>
<td>0.10 1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>A</th>
<th>C</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0.90 0</td>
<td>0</td>
<td>0</td>
<td>0.90 0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0.20 * 0</td>
<td>1</td>
<td>0.20 0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0.30 1</td>
<td>0</td>
<td>0.70 1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0.20 1</td>
<td>1</td>
<td>0.10 1</td>
<td>1</td>
</tr>
</tbody>
</table>
Weighted clauses

$\infty : A \text{ and } 2 : \neg A \vee B$

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th></th>
<th></th>
<th>0</th>
<th></th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td></td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
<td>1</td>
<td>0</td>
<td>exp(-2)</td>
<td></td>
</tr>
</tbody>
</table>

Finding the most probable instantiation (highest weighted model) is the weighted MAX-SAT problem.
Bayesian networks

If we restrict so that:

1. each variable has an associated factor,
2. this factor is a probability distribution for the variable conditional on its ‘parents’, and
3. the associated directed graph is acyclic

then we have a *Bayesian network*
Inference in propositional probabilistic models

Key problems are:

▶ Compute the marginal distribution of one or more variables.
▶ Find the instantiation with the highest probability.

Variable elimination:

▶ Can replace two factors by their product.
▶ If a variable is in only one factor then can sum it out of the distribution by summing it out of this factor.
First-order logic

First-order logic → First-order probabilistic models

↑

Propositional logic → Propositional probabilistic models
Characteristics of first-order logic

- Propositions now assert:
  - Properties of objects
  - Relations between objects
- Objects are represented by ground terms
- Can *quantify over* objects
- Universally quantified formula \( \equiv \) template for all its ground instances.

\[
\forall X : even(X) \rightarrow odd(s(X)) \vdash even(0) \rightarrow odd(s(0))
\]

\[
\forall X : even(X) \rightarrow odd(s(X)) \vdash even(s(0)) \rightarrow odd(s(s(0)))
\]

\[
\forall X, Y : p(X, Y) \land p(Y, Z) \rightarrow p(X, Z) \vdash p(a, b) \land p(b, c) \rightarrow p(a, c)
\]

\[
\forall X, Y : p(X, Y) \land p(Y, Z) \rightarrow p(X, Z) \vdash p(a, e) \land p(e, c) \rightarrow p(a, c)
\]
Factor representation of universally quantified formulae

$$\forall X, Y : p(X) \rightarrow q(X, Y)$$

<table>
<thead>
<tr>
<th>p(X)</th>
<th>q(X,Y)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>p(a)</th>
<th>q(a,b)</th>
<th></th>
<th>p(b)</th>
<th>q(b,c)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

James Cussens University of York, UK
First-order models

- Restrict attention to Herbrand models.
- Each Herbrand model assigns TRUE or FALSE to each ground atomic formula (atom for short).
- So ground atoms act like binary variables.

\[ M_1 : \quad p(a) = T, \ p(b) = F, \ q(a, b) = T, \ldots \]
\[ M_2 : \quad p(a) = T, \ p(b) = T, \ q(a, b) = F, \ldots \]
Crucially, we can stay first-order when doing inference.

Here’s an example of resolution:

\[
\forall X, Y : p(X, Y) \lor q(X), \quad \forall X, Y : \neg p(X, a) \lor r(Y, b)
\]

\[
\forall X : q(X) \lor r(a, b)
\]
First-order probabilistic models (parfactors)

First-order logic $\rightarrow$ First-order probabilistic models

Propositional logic $\rightarrow$ Propositional probabilistic models
Parfactors represent many factors with a single factor parameterised by logical variables (and constraints).

\[
\begin{array}{c|c|c}
X & Y & p(X, Y) \\
\hline
0 & 0 & 0.10 \\
0 & 1 & 0.20 \\
1 & 0 & 0.30 \\
1 & 1 & 0.20 \\
\end{array}
\]

stands for the product of all its ground instances where \( X \neq Y \):

\[
\begin{array}{c|c|c|c|c|c|c|c}
p(a) & q(a, b) & p(b) & q(b, c) & p(a) & q(a, b) & p(b) & q(b, c) \\
\hline
0 & 0 & 0 & 0.10 & 0 & 0 & 0 & 0.10 \\
0 & 1 & 0 & 0.20 & 0 & 1 & 0 & 0.20 \\
1 & 0 & 0 & 0.30 & 1 & 0 & 0 & 0.30 \\
1 & 1 & 0 & 0.20 & 1 & 1 & 0 & 0.20 \\
\end{array}
\]
What sort of probability distribution is defined?

- Each ground atom becomes a random variable.
- (If these are binary), the distribution is over Herbrand models: possible worlds.
- A parfactor with only finitely many instances can be replaced by these instances: grounding out.
- It’s just a probability distribution: don’t necessarily have to use logic to analyse/manipulate it.
Can we maintain the first-order representation when doing inference (marginalisation, maximisation) in FOPMs?

Can we implicitly sum out $p(a), p(b), \ldots$, the ground instances of $p(X)$, by summing out $p(X)$?
Lifted inference in first-order probabilistic models

▶ Can we maintain the first-order representation when doing inference (marginalisation, maximisation) in FOPMs?
▶ Can we *implicitly* sum out \( p(a), p(b), \ldots \), the ground instances of \( p(X) \), by summing out \( p(X) \)?
▶ You’re in the right place (wrong talk) to get the full story!
▶ Exploit repeated structure when it’s available.
Propositional logic
Propositional probabilistic models
First-order logic
First-order probabilistic models (parfactors)
First-order probabilistic models (generative)
SRL in statistical genetics

Markov logic parfactors

2 : $\neg A \lor B$

| A | B |  
|---|---|---|
| 0 | 0 | 1  
| 0 | 1 | 1  
| 1 | 0 | $\exp(-2)$  
| 1 | 1 | 1  

2 : $\forall X, Y : p(X, Y) \rightarrow q(X, Y)$

| p(X,Y) | q(X,Y) |  
|-------|-------|---|
| 0 | 0 | 1  
| 0 | 1 | 1  
| 1 | 0 | $\exp(-2)$  
| 1 | 1 | 1  

James Cussens University of York, UK
Let $n_i(x)$ be the number of ground instances of clause $F_i$ which hold in world $x$.

$$P(x) = Z_w^{-1} \exp \left( \sum_i w_i n_i(x) \right)$$  

(1)

An MLN defines an exponential-family distribution where $(n_1(x), n_2(x), \ldots n_k(x))$ is the canonical statistic.
What’s the data?

- The distribution is over possible worlds, so the obvious option is to assume *independent and identically distributed (iid)* instances of possible worlds.
- A single observed world—a relational database—can be enough, due to parameter sharing.
- For Markov logic networks we just need counts of true groundings.
First-order probabilistic models (generative)

First-order logic \longrightarrow \text{First-order probabilistic models}

Propositional logic \longrightarrow \text{Propositional probabilistic models}
Dynamic probabilistic models

We already quantify over random variables (sometimes implicitly) in dynamic probabilistic models:

- Time series: \( \forall t : X_t \sim aX_{t-1} + \epsilon \)
- Similarly in spatial statistics
- Stochastic grammars, including HMMs
- Dynamic Bayesian networks

A number of formalisms (e.g. ICL, SLPs, PRISM) generalise this approach.
The PRISM approach

The division of labour is:

**Probability** Very simple—families of independent and identically distributed random variables.

**Logic** Arbitrarily complex—using a standard first-order theory.
An example ‘base’ probability distribution

- Let $X_1, X_2, X_3, \ldots$ be an infinite collection of independent and identically distributed (iid) random variables taking values ‘y’ and ‘n’. Suppose $P(X_i = y) = 0.3$.
- Similarly let $Y_1, Y_2, Y_3, \ldots$ and $Z_1, Z_2, Z_3, \ldots$ also be iid families with values in $\{0, 1\}$ and $\{0, 1, 2, \ldots 9\}$, respectively.
- Here’s (the beginning) of a joint instantiation of all these variables:

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X$</td>
<td>y</td>
<td>n</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>n</td>
<td>...</td>
</tr>
<tr>
<td>$Y$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>...</td>
</tr>
<tr>
<td>$Z$</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>5</td>
<td>2</td>
<td>...</td>
</tr>
</tbody>
</table>
Defining a ‘base’ distribution in PRISM

Here’s the PRISM source defining the example distribution:

```prolog
values('X', [y,n]).
values('Y', [0,1]).
values('Z', [0,1,2,3,4,5,6,7,8,9]).

:- set_sw('X', 0.3+0.7).
:- set_sw('Y', 0.4+0.6).
:- set_sw('Z', 0.1+0.1+0.1+0.1+0.1+0.1+0.1+0.1+0.1).
```

James Cussens University of York, UK
A joint instantiation determines a logical theory

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>y</td>
<td>n</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>n</td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Z</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>5</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

This joint instantiation determines the following logical theory:

\[
\text{msw}(\text{'X', 1, y}), \text{msw}(\text{'X', 2, n}), \text{msw}(\text{'X', 3, y}), \ldots \\
\text{msw}(\text{'Y', 1, 0}), \text{msw}(\text{'Y', 2, 1}), \text{msw}(\text{'Y', 3, 0}), \ldots \\
\text{msw}(\text{'Z', 1, 4}), \text{msw}(\text{'Z', 2, 3}), \text{msw}(\text{'Z', 3, 1}), \ldots 
\]
Using a fixed, arbitrary logical theory to extend a base distribution

- Can extend this simple base distribution by considering what becomes true once a probabilistically chosen theory is added to an existing fixed logical theory $R$.

- Let $\text{fla}$ be some first-order sentence. $\text{Prob}(\text{fla})$ is the probability of getting a base joint instantiation $F$ such that $F, R \vdash \text{fla}$.

- Use Closed World Assumption to get this to work.
Working with target predicates

It is convenient to specify a *target predicate* such that exactly one ground atom with this predicate symbol follows from any choice of $F$.

- $F_1, R \vdash t(a)$
- $F_2, R \vdash t(b)$
- $F_3, R \vdash t(a)$

▶ Defines a distribution over the *success* set of $t$.
▶ This can be generalised to allow *at most one* target ground atom to follow (*failure* models).
Computing target probabilities from a PRISM distribution

- We don’t consider all possible infinite instantiations of the base distribution!
- It’s a PRISM requirement that $\text{Prob}(t(a))$ for any target atom $t(a)$ is a finite sum of finite products of base distribution probabilities.
- For a given $t(a)$, *abduction* is used to find (conjunctions of) ‘msw’ facts that make $t(a)$ true.
Abduction: a HMM example

\[
\text{hmm}([a,b,a]) \\
\iff \text{msw}(\text{out}(s0),1,a) \land \text{msw}(\text{tr}(s0),1,s0) \land \text{hmm}(s0,[b,a]) \\
\lor \text{msw}(\text{out}(s0),1,a) \land \text{msw}(\text{tr}(s0),1,s1) \land \text{hmm}(s1,[b,a])
\]
Abduction: a HMM example

\[
\text{hmm}([a,b,a]) \\
\iff \text{msw}(\text{out}(s0),1,a) \land \text{msw}(\text{tr}(s0),1,s0) \land \text{hmm}(s0,[b,a]) \\
\lor \text{msw}(\text{out}(s0),1,a) \land \text{msw}(\text{tr}(s0),1,s1) \land \text{hmm}(s1,[b,a]) \\
\text{hmm}(s0,[b,a]) \\
\iff \text{msw}(\text{out}(s0),2,b) \land \text{msw}(\text{tr}(s0),2,s0) \land \text{hmm}(s0,[a]) \\
\lor \text{msw}(\text{out}(s0),2,b) \land \text{msw}(\text{tr}(s0),2,s1) \land \text{hmm}(s1,[a])
\]
Abduction: a HMM example

\[
hmm([a,b,a]) \\
ü \iff msw(out(s0),1,a) \land msw(tr(s0),1,s0) \land hmm(s0,[b,a]) \\
\lor msw(out(s0),1,a) \land msw(tr(s0),1,s1) \land hmm(s1,[b,a]) \\
hmm(s0,[b,a]) \\
\iff msw(out(s0),2,b) \land msw(tr(s0),2,s0) \land hmm(s0,[a]) \\
\lor msw(out(s0),2,b) \land msw(tr(s0),2,s1) \land hmm(s1,[a]) \\
hmm(s1,[b,a]) \\
\iff msw(out(s1),1,b) \land msw(tr(s1),1,s0) \land hmm(s0,[a]) \\
\lor msw(out(s1),1,b) \land msw(tr(s1),1,s1) \land hmm(s1,[a])
\]
Abduction: a HMM example

\[
\text{hmm}([a,b,a]) \\
\Leftrightarrow \text{msw(out}(s0),1,a) \land \text{msw(tr}(s0),1,s0) \land \text{hmm}(s0,[b,a]) \\
\lor \text{msw(out}(s0),1,a) \land \text{msw(tr}(s0),1,s1) \land \text{hmm}(s1,[b,a]) \\
\text{hmm}(s0,[b,a]) \\
\Leftrightarrow \text{msw(out}(s0),2,b) \land \text{msw(tr}(s0),2,s0) \land \text{hmm}(s0,[a]) \\
\lor \text{msw(out}(s0),2,b) \land \text{msw(tr}(s0),2,s1) \land \text{hmm}(s1,[a]) \\
\text{hmm}(s1,[b,a]) \\
\Leftrightarrow \text{msw(out}(s1),1,b) \land \text{msw(tr}(s1),1,s0) \land \text{hmm}(s0,[a]) \\
\lor \text{msw(out}(s1),1,b) \land \text{msw(tr}(s1),1,s1) \land \text{hmm}(s1,[a]) \\
\text{hmm}(s0,[a]) \\
\Leftrightarrow \text{msw(out}(s0),3,a) \land \text{msw(tr}(s0),3,\text{stop})
\]
Abduction: a HMM example

\[
\begin{align*}
\text{hmm}([a,b,a]) & \iff 
\text{msw}(\text{out}(s0),1,a) \land \text{msw}(\text{tr}(s0),1,s0) \land \text{hmm}(s0, [b,a]) \\
& \land 
\text{msw}(\text{out}(s1),1,a) \land \text{msw}(\text{tr}(s1),1,s0) \land \text{hmm}(s0, [a]) \\
& \lor 
\text{msw}(\text{out}(s0),2,b) \land \text{msw}(\text{tr}(s0),2,s0) \land \text{hmm}(s0, [a]) \\
& \land 
\text{msw}(\text{out}(s1),2,b) \land \text{msw}(\text{tr}(s1),2,s1) \land \text{hmm}(s1, [b,a]) \\
& \lor 
\text{msw}(\text{out}(s0),3,a) \land \text{msw}(\text{tr}(s0),3,\text{stop}) \land \text{hmm}(s1, [a]) \\
& \iff 
\text{msw}(\text{out}(s1),2,a) \land \text{msw}(\text{tr}(s1),2,\text{stop}) \\
& \text{hmm}(s0, [a]) \\
& \lor 
\text{msw}(\text{out}(s1),2,a) \land \text{msw}(\text{tr}(s1),2,\text{stop}) \land \text{hmm}(s1, [a])
\end{align*}
\]
Computing probabilities by abduction

\[
\text{hmm}([a,b,a])
\iff \text{msw}(\text{out}(s0),1,a) \land \text{msw}(\text{tr}(s0),1,s0) \land \text{hmm}(s0,[b,a]) \\
\lor \text{msw}(\text{out}(s0),1,a) \land \text{msw}(\text{tr}(s0),1,s1) \land \text{hmm}(s1,[b,a])
\]

\[
\text{hmm}(s0,[b,a])
\iff \text{msw}(\text{out}(s0),2,b) \land \text{msw}(\text{tr}(s0),2,s0) \land \text{hmm}(s0,[a]) \\
\lor \text{msw}(\text{out}(s0),2,b) \land \text{msw}(\text{tr}(s0),2,s1) \land \text{hmm}(s1,[a])
\]

\[
\text{hmm}(s1,[b,a])
\iff \text{msw}(\text{out}(s1),1,b) \land \text{msw}(\text{tr}(s1),1,s0) \land \text{hmm}(s0,[a]) \\
\lor \text{msw}(\text{out}(s1),1,b) \land \text{msw}(\text{tr}(s1),1,s1) \land \text{hmm}(s1,[a])
\]

\[
\text{hmm}(s0,[a])
\iff \text{msw}(\text{out}(s0),3,a) \land \text{msw}(\text{tr}(s0),3,\text{stop})
\]

\[
\text{hmm}(s1,[a])
\iff \text{msw}(\text{out}(s1),2,a) \land \text{msw}(\text{tr}(s1),2,\text{stop})
\]
Computing probabilities by abduction

\[
\Pr(\text{hmm}([a,b,a])) \\
= \Pr(\text{msw}(\text{out}(s0),1,a)) \times \Pr(\text{msw}(\text{tr}(s0),1,s0)) \times \Pr(\text{hmm}(s0,[b,a])) \\
+ \Pr(\text{msw}(\text{out}(s0),1,a)) \times \Pr(\text{msw}(\text{tr}(s0),1,s1)) \times \Pr(\text{hmm}(s1,[b,a]))
\]

\[
\text{hmm}(s0,[b,a]) \iff \text{msw}(\text{out}(s0),2,b) \land \text{msw}(\text{tr}(s0),2,s0) \land \text{hmm}(s0,[a]) \\
\lor \text{msw}(\text{out}(s0),2,b) \land \text{msw}(\text{tr}(s0),2,s1) \land \text{hmm}(s1,[a])
\]

\[
\text{hmm}(s1,[b,a]) \iff \text{msw}(\text{out}(s1),1,b) \land \text{msw}(\text{tr}(s1),1,s0) \land \text{hmm}(s0,[a]) \\
\lor \text{msw}(\text{out}(s1),1,b) \land \text{msw}(\text{tr}(s1),1,s1) \land \text{hmm}(s1,[a])
\]

\[
\text{hmm}(s0,[a]) \iff \text{msw}(\text{out}(s0),3,a) \land \text{msw}(\text{tr}(s0),3,\text{stop})
\]

\[
\text{hmm}(s1,[a]) \iff \text{msw}(\text{out}(s1),2,a) \land \text{msw}(\text{tr}(s1),2,\text{stop})
\]
Computing probabilities by abduction

\[
\begin{align*}
\Pr(\text{hmm}([a,b,a])) &= \Pr(\text{msw}(\text{out}(s0),1,a)) \times \Pr(\text{msw}(\text{tr}(s0),1,s0)) \times \Pr(\text{hmm}(s0,[b,a])) \\
&\quad + \Pr(\text{msw}(\text{out}(s0),1,a)) \times \Pr(\text{msw}(\text{tr}(s0),1,s1)) \times \Pr(\text{hmm}(s1,[b,a])) \\
\Pr(\text{hmm}(s0,[b,a])) &= \Pr(\text{msw}(\text{out}(s0),2,b)) \times \Pr(\text{msw}(\text{tr}(s0),2,s0)) \times \Pr(\text{hmm}(s0,[a])) \\
&\quad + \Pr(\text{msw}(\text{out}(s0),2,b)) \times \Pr(\text{msw}(\text{tr}(s0),2,s1)) \times \Pr(\text{hmm}(s1,[a])) \\
\text{hmm}(s1,[b,a]) &\Leftrightarrow \text{msw}(\text{out}(s1),1,b) \land \text{msw}(\text{tr}(s1),1,s0) \land \text{hmm}(s0,[a]) \\
&\lor \text{msw}(\text{out}(s1),1,b) \land \text{msw}(\text{tr}(s1),1,s1) \land \text{hmm}(s1,[a]) \\
\text{hmm}(s0,[a]) &\Leftrightarrow \text{msw}(\text{out}(s0),3,a) \land \text{msw}(\text{tr}(s0),3,\text{stop}) \\
\text{hmm}(s1,[a]) &\Leftrightarrow \text{msw}(\text{out}(s1),2,a) \land \text{msw}(\text{tr}(s1),2,\text{stop})
\end{align*}
\]
Computing probabilities by abduction

\[
\Pr(\text{hmm}([a,b,a])) = \Pr(\text{msw}(\text{out}(s0),1,a)) \times \Pr(\text{msw}(\text{tr}(s0),1,s0)) \times \Pr(\text{hmm}(s0,[b,a])) \\
+ \Pr(\text{msw}(\text{out}(s0),1,a)) \times \Pr(\text{msw}(\text{tr}(s0),1,s1)) \times \Pr(\text{hmm}(s1,[b,a])) \\
\Pr(\text{hmm}(s0,[b,a])) = \Pr(\text{msw}(\text{out}(s0),2,b)) \times \Pr(\text{msw}(\text{tr}(s0),2,s0)) \times \Pr(\text{hmm}(s0,[a])) \\
+ \Pr(\text{msw}(\text{out}(s0),2,b)) \times \Pr(\text{msw}(\text{tr}(s0),2,s1)) \times \Pr(\text{hmm}(s1,[a])) \\
\Pr(\text{hmm}(s1,[b,a])) = \Pr(\text{msw}(\text{out}(s1),1,b)) \times \Pr(\text{msw}(\text{tr}(s1),1,s0)) \times \Pr(\text{hmm}(s0,[a])) \\
+ \Pr(\text{msw}(\text{out}(s1),1,b)) \times \Pr(\text{msw}(\text{tr}(s1),1,s1)) \times \Pr(\text{hmm}(s1,[a])) \\
\text{hmm}(s0,[a]) \iff \text{msw}(\text{out}(s0),3,a) \land \text{msw}(\text{tr}(s0),3,\text{stop}) \\
\text{hmm}(s1,[a]) \iff \text{msw}(\text{out}(s1),2,a) \land \text{msw}(\text{tr}(s1),2,\text{stop})
\]
Computing probabilities by abduction

\[
\begin{align*}
\Pr(\text{hmm}([a,b,a])) &= \Pr(\text{msw}(\text{out}(s0),1,a)) \times \Pr(\text{msw}(\text{tr}(s0),1,s0)) \times \Pr(\text{hmm}(s0,[b,a])) \\
&+ \Pr(\text{msw}(\text{out}(s0),1,a)) \times \Pr(\text{msw}(\text{tr}(s0),1,s1)) \times \Pr(\text{hmm}(s1,[b,a])) \\
\Pr(\text{hmm}(s0,[b,a])) &= \Pr(\text{msw}(\text{out}(s0),2,b)) \times \Pr(\text{msw}(\text{tr}(s0),2,s0)) \times \Pr(\text{hmm}(s0,[a])) \\
&+ \Pr(\text{msw}(\text{out}(s0),2,b)) \times \Pr(\text{msw}(\text{tr}(s0),2,s1)) \times \Pr(\text{hmm}(s1,[a])) \\
\Pr(\text{hmm}(s1,[b,a])) &= \Pr(\text{msw}(\text{out}(s1),1,b)) \times \Pr(\text{msw}(\text{tr}(s1),1,s0)) \times \Pr(\text{hmm}(s0,[a])) \\
&+ \Pr(\text{msw}(\text{out}(s1),1,b)) \times \Pr(\text{msw}(\text{tr}(s1),1,s1)) \times \Pr(\text{hmm}(s1,[a])) \\
\Pr(\text{hmm}(s0,[a])) &= \Pr(\text{msw}(\text{out}(s0),3,a)) \times \Pr(\text{msw}(\text{tr}(s0),3,\text{stop})) \\
\text{hmm}(s1,[a]) \Leftrightarrow \text{msw}(\text{out}(s1),2,a) \land \text{msw}(\text{tr}(s1),2,\text{stop})
\end{align*}
\]
Computing probabilities by abduction

\[
\begin{align*}
\Pr(\text{hmm}([a,b,a])) &= \Pr(\text{msw}(\text{out}(s0),1,a)) \times \Pr(\text{msw}(\text{tr}(s0),1,s0)) \times \Pr(\text{hmm}(s0,[b,a])) \\
&\quad + \Pr(\text{msw}(\text{out}(s0),1,a)) \times \Pr(\text{msw}(\text{tr}(s0),1,s1)) \times \Pr(\text{hmm}(s1,[b,a])) \\
\Pr(\text{hmm}(s0,[b,a])) &= \Pr(\text{msw}(\text{out}(s0),2,b)) \times \Pr(\text{msw}(\text{tr}(s0),2,s0)) \times \Pr(\text{hmm}(s0,[a])) \\
&\quad + \Pr(\text{msw}(\text{out}(s0),2,b)) \times \Pr(\text{msw}(\text{tr}(s0),2,s1)) \times \Pr(\text{hmm}(s1,[a])) \\
\Pr(\text{hmm}(s1,[b,a])) &= \Pr(\text{msw}(\text{out}(s1),1,b)) \times \Pr(\text{msw}(\text{tr}(s1),1,s0)) \times \Pr(\text{hmm}(s0,[a])) \\
&\quad + \Pr(\text{msw}(\text{out}(s1),1,b)) \times \Pr(\text{msw}(\text{tr}(s1),1,s1)) \times \Pr(\text{hmm}(s1,[a])) \\
\Pr(\text{hmm}(s0,[a])) &= \Pr(\text{msw}(\text{out}(s0),3,a)) \times \Pr(\text{msw}(\text{tr}(s0),3,\text{stop})) \\
\Pr(\text{hmm}(s1,[a])) &= \Pr(\text{msw}(\text{out}(s1),2,a)) \times \Pr(\text{msw}(\text{tr}(s1),2,\text{stop}))
\end{align*}
\]
What’s the data?

- In PRISM we assume iid worlds, but we only see the instance of the target predicate true in each world
- Thus have to use EM to fit parameters
Bayesian network learning for pedigrees

- At most 2 parents per child (hurray!)
- Parameters are known, pure structure learning
- Lots of logical constraints
- Relations between objects
Some genetics

- Different variants of a gene are called *alleles*.
- You get one allele from your mother and one from your father: your *genotype*.
- Typically one observes *unordered genotypes*: don’t know parental origin of the allele.
The problem

Given

- A set $G$ of possible pedigrees;
- a prior over pedigrees $P(g)$;
- observed marker data $x_o$;
- and an assumption of Mendelian segregation

Find

- $\arg \max_{g \in G} P(g| x_o)$
Defining a joint probability distribution

- Good Bayesians reduce learning to probabilistic inference.
- There will be four disjoint collections of binary variables to encode:
  1. the pedigree \( (g) \)
  2. the unobserved ordered genotypes \( (y) \)
  3. the observed and unobserved unordered genotypes \( (x = (x_h, x_o)) \)
  4. and the (possibly observed) auxiliary variables giving e.g. relative age information \( (z) \).
- An exponential-family distribution will be defined for \( P(g, x, y, z) \) using Markov logic.
Pedigree and auxiliary variables

Pedigree variables  father(bob, alice), mother(alice, rob), . . .
Auxiliary variables  older(bob, alice), . . .

There are many constraints, for example:

- $\forall X, Y : \text{father}(X, Y) \rightarrow \text{older}(X, Y)$,
  $\forall X, Y, Z : \text{older}(X, Y) \land \text{older}(Y, Z) \rightarrow \text{older}(X, Z), . . .$
- $\forall X, Y, Z : X \neq Y \rightarrow \neg \text{father}(X, Z) \lor \neg \text{father}(Y, Z), . . .$
Ordered genotype variables pat(bob, a2), mat(alice, a4), ...

- **pat** and **mat** are functional relations:
  \[
  \forall X, A, B : A \neq B \rightarrow \neg \text{mat}(X, A) \lor \neg \text{mat}(X, B),
  \]
  \[
  \forall X : \exists A : \text{pat}(X, A), \ldots
  \]

- **Homozygous inheritance**:
  \[
  \forall X, Y, A : \text{pat}(X, A) \land \text{mat}(X, A) \land \text{father}(X, Y) \rightarrow \text{pat}(Y, A).
  \]
Unordered genotype variables

- $\forall X, A, B : \text{genotype}(X, A, B) \leftrightarrow (\text{pat}(X, A) \land \text{mat}(X, B)) \lor (\text{mat}(X, A) \land \text{pat}(X, B))$
As is typical only true ground atoms are listed:

father(m1,m2)    pa(m2,a1)
older(m1,m2)  pa(f1,a3)
mother(f1,m1)  ma(m1,a1)
older(f1,m1) ma(m2,a2)
mother(f1,m2) ma(f1,a1)
older(f1,m2) genotype(m1,a1,a1)
pa(m1,a1) genotype(m2,a1,a2)
pa(m1,a1) genotype(f1,a1,a3)
Penalty for heterozygosity

Assuming Mendelian segregation

\[ - \log 0.5 : \forall X, Y, A, B : \]
\[ \neg (\text{father}(X, Y) \land \text{pat}(X, A) \land \text{mat}(X, B) \land A \neq B) \]
Encoding population frequencies

\[-\log p_i : \forall Y : \exists X : \text{father}(X, Y) \lor \neg \text{pat}(Y, a_i)\]

\[-\log p_i : \forall Y : \exists X : \text{mother}(X, Y) \lor \neg \text{mat}(Y, a_i)\]
Priors on pedigrees

For example:

30 : $\forall X, Y, Z : mother(X, Y) \land father(Y, Z) \rightarrow \neg mother(X, Z)$

20 : $\forall X, Y, Z : mother(X, Z) \land father(Y, Z) \rightarrow \neg related(X, Y)$

Just a different way of writing down the prior of (Sheehan & Egeland, 2007).
Incorporating evidence

- Just add in the appropriate ground atoms, e.g.
  - genotype(bob, a1, a2)
  - father(john, robin)
- thus ruling out all worlds in which these are not true.
- The intelligent approach is to ‘propagate’ the evidence to specialise the general-purpose logical knowledge base.
An simple example

With a uniform prior on pedigrees and this (unordered) genotype data:

```
genotype(m1,a1,a2)  genotype(f1,a2,a2)
genotype(m2,a1,a2)  genotype(f2,a1,a2)
genotype(m3,a2,a2)  genotype(f3,a2,a2)
genotype(m4,a1,a2)  genotype(f4,a1,a2)
genotype(m5,a1,a1)  genotype(f5,a1,a1)
mother(f1,f3)
```
Grounding out and using the exact weighted MAX-SAT solver (minimaxsat1.0, Heras et al) took 30 seconds to establish that this ‘possible world’ is the most probable:

father(m2,m4)  mother(f1,m1)  pa(m1,a1)  ma(m1,a2)
father(m2,f1)  mother(f1,m3)  pa(m2,a1)  ma(m2,a2)
father(m2,f5)  mother(f1,m4)  pa(m3,a2)  ma(m3,a2)
father(m4,m1)  mother(f1,f3)  pa(m4,a1)  ma(m4,a2)
father(m4,m3)  mother(f2,m2)  pa(m5,a1)  ma(m5,a1)
father(m4,m5)  mother(f2,f1)  pa(f1,a2)  ma(f1,a2)
father(m4,f3)  mother(f2,f4)  pa(f2,a1)  ma(f2,a2)
father(m4,f4)  mother(f2,f5)  pa(f3,a2)  ma(f3,a2)
father(m4,f5)  mother(f5,m5)  pa(f4,a2)  ma(f4,a1)
father(m4,f5)  mother(f5,m5)  pa(f5,a1)  ma(f5,a1)
Another result

Took 145s.

```
  genotype(m1,a1,a2)  genotype(f1,a2,a2)
  genotype(m2,a1,a2)  genotype(f2,a1,a2)
  genotype(m3,a2,a2)  genotype(f3,a2,a2)
  genotype(m4,a1,a2)  genotype(f4,a1,a2)
  genotype(m5,a1,a1)  genotype(f5,a1,a1)
```

If a total order is added, this reduces to 0.076s.
This is a nice way of solving $\arg \max_{g,y} P(x, y|g)P(g)$, but we actually want to solve
$\arg \max_g P(x|g)P(g) = \arg \max_g \sum_y P(x, y|g)P(g)$.

Domingos’s group (University of Washington) working on this right now.