Outline

- Why probabilistic logic?
- Abduction as an underlying framework
- Abduction by logic programming
- Relationship between graphical models and probabilistic logic
- Research topics
Why Relational + Probabilistic

- **Structure**, but also **uncertainty**
- Medical diagnosis, pedigrees, etc.
- Social network structures, citation analysis, etc.
Representations

- We have seen generalisations of propositional logic:

\[ \text{Talented} \land \text{GoodTeacher} \rightarrow \text{PassCourse} \]

Extensions in different directions:

1. How to incorporate uncertainty?
   - Rule-based uncertainty (1970’s and 1980’s)
     \[ \text{Talented} \land \text{GoodTeacher} \xrightarrow{\text{CF} = 0.9} \text{PassCourse} \]
   - Bayesian networks (1990’s – now)

2. How to incorporate relations?
   - First-order logic as a general language
Bayesian networks

\[ P(p \mid t, g) = 0.9 \]
\[ P(p \mid \neg t, g) = 0.5 \]
\[ P(p \mid t, \neg g) = 0.7 \]
\[ P(p \mid \neg t, \neg g) = 0.1 \]
\[ P(g) = 0.6 \]
\[ P(t) = 0.9 \]

\[ P(P, S, T) = P(P \mid S, T)P(S)P(T) \]

Allows computing arbitrary probabilities:

\[ P(p \mid t, g) = 0.9 \]
\[ P(p \mid t) = \ldots \]
\[ P(t \mid p) = \ldots \]
First-order logic

First-order logic allows modelling relations; consider this formula $\varphi$:

$$\forall s \text{Talented}(s) \land \text{GoodTeacher} \to \text{PassCourse}(s) \quad (1)$$

Example reasoning (first-order, abductive):

- If John does not pass the course, then (obviously) it is because of the teachers

$$\{\varphi, \forall s \text{Talented}(s), \neg \text{PassCourse}(J)\} \models \neg \text{GoodTeacher}$$

- This might be bad news for Mary, because now there is no hypothesis $H$ such that:

$$\{\varphi, \forall s \text{Talented}(s), \neg \text{PassCourse}(J), H\} \models \text{PassCourse}(M)$$
Probability and First-Order Domains

- FOL can talk about **constants** and **relations**
- Propositional/BN only about **fixed** settings
- So, FOL really opens up new possibilities for representing, reasoning and learning, but...
- We have seen before (with CF) that naive combinations of logic and probability need to be approached with care
- A key general issue: what does it mean to say:
  - \( P(\text{Some randomly chosen bird can fly}) \geq 0.8 \)
  - \( P(\text{Tweety can fly}) \geq 0.8 \) (Tweety is a particular bird)
- Halpern (1990): type-1 and type-2 probability
Probabilistic relational reasoning

- First-order logic: good for *relational reasoning in various ways* about classes of objects
- Probabilistic graphical models such as Bayesian networks are *good for reasoning with uncertainty*

⇒ *Is there no way to combine them?*

Solutions for:
- Probability that all students are talented
- Probability that Mary will pass the course, given the observations about John
Many combinations

- Hot topic in AI, many approaches since end of the nineties
- Start from logic programming: KBMC, SLP, PRISM, LPAD, ICL, CPLlogic, SCFG, etc.
- Start from probabilistic graphical models: BLP, BLN, SRM, RMM, MLN, RDN, etc.

Probabilistic programming languages starting to appear
- Imperative-Functional: infer.net, Church, Factorie, Scala, etc.
- Horn-Logic and Prolog: PRISM, ProbLog, ICL, Dyna, etc.
Lessons learned

Consider:

- if $P(a) = 0.3$ and $P(b) = 0.6$, what is $P(a \land b)$?
- if $P(a) = 0.3$ and $P(b) = 0.6$, what is $P(a \lor b)$?
- if $P(h \mid e) = 0.3$ and $P(h \mid e') = 0.3$, what is $P(h \mid e, e')$?
Probabilistic reasoning

$P(x_3) = \sum_{X_1, X_2} P(x_3 \mid X_1, X_2) P(X_1) P(X_2)$

$P(x_4 \mid x_3) = 0.4$
$P(x_4 \mid \neg x_3) = 0.1$
$P(x_3 \mid x_1, x_2) = 0.3$
$P(x_3 \mid \neg x_1, x_2) = 0.5$
$P(x_3 \mid x_1, \neg x_2) = 0.7$
$P(x_3 \mid \neg x_1, \neg x_2) = 0.9$
$P(x_1) = 0.6$
$P(x_2) = 0.2$

probabilistic reasoning = abduction?
Recall: abductive explanations

Causal specification: $\Sigma = (\Delta, \Phi, \mathcal{R})$, met:

- $\Delta$: potential causes and incompleteness assumptions (assumables)
- $\Phi$: facts that can be observed
- $\mathcal{R}$: causal model

Explanations (prediction) $E \subseteq \Delta$: $\mathcal{R} \cup E \models F$

Let $\mathcal{E}(F)$ be the set of all explanations of $F$
Example explanations

Causal specification: \( \Sigma = (\Delta, \Phi, \mathcal{R}) \)

- Example 1: \( \mathcal{R} \cup \{flu, \alpha_1\} \models chills \land thirst \)
- Example 2: \( \mathcal{R} \cup \{flu, \alpha_1, \alpha_2\} \models chills \land thirst \)

The set of all explanations for chills and thirst contains:

\[
\mathcal{E}(chills \land thirst) = \{\{flu, \alpha_1\}, \{flu, \alpha_1, \alpha_2\}, \\
\{flu, \alpha_1, sport\}, \{flu, \alpha_1, sport, \alpha_2\}\}
\]
Closed world assumption: $F$ is only true if and only if one of its explanations is true:

$$F = \bigvee_{E_i \in \mathcal{E}(F)} E_i$$

E.g.: $\text{chills} \land \text{thirst} = (\text{flu} \land \alpha_1) \lor (\text{flu} \land \alpha_1 \land \alpha_2) \\
\lor (\text{flu} \land \alpha_1 \land \text{sport}) \lor (\text{flu} \land \alpha_1 \land \text{sport} \land \alpha_2)$
Idea for adding probabilities

Suppose we have a probability distribution over $\Delta$, i.e., $P(\Delta)$, then we can compute $P(F)$, because:

$$P(F) = P(\bigvee_{E_i \in \mathcal{E}(F)} E_i)$$

$$P(\text{chills} \land \text{thirst}) = P((\text{flu} \land \alpha_1) \lor (\text{flu} \land \alpha_1 \land \alpha_2) \lor (\text{flu} \land \alpha_1 \land \text{sport}) \lor (\text{flu} \land \alpha_1 \land \text{sport} \land \alpha_2))$$
Definition. A minimal explanation $E$ for $F$ is an explanation $E$ for $F$ s.t. there is no $E' \subset E$ where $E'$ is an explanation for $F$.

Theorem. Let $\mathcal{E}_m(F)$ be the set of all minimal explanations for $F$. Then:

$$F = \bigvee_{E_i \in \mathcal{E}_m(F)} E_i$$

Proof (sketch). Note that if $E_i \in \mathcal{E}_m(F)$ and $E_j \supset E_i$, then:

$$E_i \lor E_j = E_i$$

Proof by induction on the number of non-minimal explanations
**Minimal explanations: example**

\[
\begin{align*}
\text{chills} \land \text{thirst} & = (\text{flu} \land \alpha_1) \lor (\text{flu} \land \alpha_1 \land \text{sport}) \lor \cdots \\
& = \text{flu} \land \alpha_1
\end{align*}
\]

Recall that this is the solution formula \( S \) for \( F \): the most specific formula consisting only of abducible literals, such that

\[
COMP[\mathcal{R}; N] \cup F \models S
\]
Defining a probability distribution

We assume a very simple distribution consisting of a set of independent random variables

Partition $V \subseteq \Delta$ is associated to a random variable $X_V$ where $V$ is the domain of $X$

Example:

$P(X = \text{sport}) = 0.3$  
$P(X = \text{flu}) = 0.1$  
$P(X = \text{not\_sport\_or\_flu}) = 0.6$  
$P(Y = \alpha_1) = 0.9$  
$P(Y = \text{other}_1) = 0.1$  
$P(Z = \alpha_2) = 0.7$  
$P(Z = \text{other}_2) = 0.3$

(Assumption: sport and flu are mutually exclusive)
Example

\[
P(\text{myalgia}) = P((\text{flu} \land \alpha_2) \lor \text{sport})
\]
\[
= P(\text{flu} \land \alpha_2) + P(\text{sport})
\]
\[
= P(\text{flu})P(\alpha_2) + P(\text{sport})
\]
\[
= 0.1 \cdot 0.7 + 0.3 = 0.37
\]

Problem: how to obtain the minimal explanations?
Recall: logic programming

- A substitution $\theta$ is a finite set of the form $\theta = \{ t_1/x_1, \ldots, t_n/x_n \}$, with $x_i$ a variable and $t_i$ a term; $x_i \neq t_i$ and $x_i \neq x_j$, $i \neq j$

- A grounded expression does not contain variables

- A substitution $\theta$ is called a unifier of $E$ and $E'$ if $E\theta = E'\theta$; $E$ and $E'$ are then called unifiable

- SLD resolution (for Horn clauses):

  $\leftarrow (B_1, \ldots, B_n)\theta, \quad (B_i \leftarrow A_1, \ldots, A_m)\theta$

  $\leftarrow (B_1, \ldots, B_{i-1}, A_1, \ldots, A_m, B_{i+1}, \ldots, B_n)\theta$

  such that $B_i$ unifies given substitution $\theta$

SLD derivation = backward reasoning + unification
## Explanations: by resolution

Given the following specification:

<table>
<thead>
<tr>
<th>Condition</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>chills ← fever</td>
<td></td>
</tr>
<tr>
<td>thirst ← fever</td>
<td></td>
</tr>
<tr>
<td>fever ← flu</td>
<td></td>
</tr>
<tr>
<td>myalgia ← flu</td>
<td></td>
</tr>
<tr>
<td>myalgia ← sport</td>
<td></td>
</tr>
</tbody>
</table>

Suppose \( F = \text{myalgia} \):

\[
\begin{array}{c}
\hline
\text{myalgia} & \text{myalgia} \leftarrow \text{sport} \\
\hline
\text{myalgia} \leftarrow \text{sport} & \text{sport} : 0.3 \\
\end{array}
\]

\[]
Explanations: by resolution

Given the following specification:

\[

chills \leftarrow fever \\
thirst \leftarrow fever \\
fever \leftarrow flu \\
myalgia \leftarrow flu \\
myalgia \leftarrow sport
\]

Suppose \( F = myalgia \):

\[
\begin{align*}
\leftarrow myalgia \quad & myalgia \leftarrow flu, \alpha_2 \\
\leftarrow flu, \alpha_2 \quad & flu : 0.1 \\
\leftarrow \alpha_2 & \alpha_2 : 0.7
\end{align*}
\]
The AILog system

prob flu : 0.1, sport : 0.3, dummy : 0.6.
prob a1 : 0.9.
prob a2 : 0.7.

chills <- fever & a1.
fever <- flu.
thirst <- fever.
myalgia <- flu & a2.
myalgia <- sport.

gives:
ailog: predict myalgia.
Answer: P(myalgia|Obs)=0.37.

[ok,more,explanations,worlds,help]: explanations.
0: ass([],[a2,flu],0.06999999999999999)
1: ass([],[sport],0.3)
The first-order case

- Explanations are sets of ground assumables
- In particular: ground assumables used in an SLD proof
- A declaration:

\[ a_1 : p, a_2 : p_2, \ldots, a_n : p_n \]

now defines a random variable \( X_i \) for every grounding of \( a_1, \ldots, a_n \) such that \( P(X_i = a_j \theta_i) = p_j \)

Example:

\[ Flu(p) : 0.1, Sport(p) : 0.3, Other(p) : 0.6 \]

implies e.g. \( Flu(Arjen) = 0.1 \)
First-order inference: example

Given is:

\[
\text{PassCourse}(s) \leftarrow \alpha(s), \text{GoodTeacher}
\]

\[
\text{GoodTeacher} : 0.7
\]

\[
\alpha(s) : 0.9
\]

What is \( P(\text{PassCourse}(M)) \)?

\[
\begin{align*}
\leftarrow \text{PassCourse}(M) & \quad \text{PassCourse}(s) \leftarrow \alpha(s), \text{GoodTeacher} \\
& \quad \leftarrow \alpha(M), \text{GoodTeacher} \\
& \quad \leftarrow \text{GoodTeacher} \quad \alpha(s) : 0.9 \quad \text{GoodTeacher} : 0.7
\end{align*}
\]

\[
P(\text{PassCourse}(M)) = P(\alpha(M) \wedge \text{GoodTeacher}) = 0.63
\]
**Conditional probabilities**

By the definition of conditional probability:

\[
P(A \mid B) = \frac{P(A \land B)}{P(B)} = \frac{P(\bigvee_{E_i \in E_m(A \land B)} E_i)}{P(\bigvee_{E_i \in E_m(B)} E_i)}
\]

Example:

\[
E_m(\text{PassCourse}(J)) = \{\{\text{GoodTeacher}, \alpha(J)\}\}
\]

\[
E_m(\text{PassCourse}(M) \land \text{PassCourse}(J)) = \{\{\text{GoodTeacher}, \alpha(J), \alpha(M)\}\}
\]

\[
P(\text{PC}(M) \mid \text{PC}(J)) = \frac{P(\text{GoodTeacher}) \cdot P(\alpha(J)) \cdot P(\alpha(M))}{P(\text{GoodTeacher}) \cdot P(\alpha(J))} = \frac{P(\alpha(M))}{P(\alpha(J))} = 0.9
\]
Expressiveness

- Every Bayesian network can be translated to probabilistic logic (*Assignment 2*)
  - Intuition: each variable given its parent in the graph becomes an implication

- Every ground probabilistic program can be converted into a Bayesian network
  - What happens to multiple rules with the same head?

- Every non-ground probabilistic program can be seen as a template for a Bayesian network:

\[
A(x) \leftarrow \alpha(x), B(x)
\]

can be seen as a piece of Bayesian network for every instantiation for \(x\)
Recall: causal independence

\[
P(e \mid C_1, \ldots, C_n) = \sum_{I_1,\ldots,I_n} P(e \mid I_1, \ldots, I_n) \prod_{k=1}^{n} P(I_k \mid C_k)
\]

\[
= \sum_{b(I_1,\ldots,I_n)=e} \prod_{k=1}^{n} P(I_k \mid C_k)
\]

Boolean functions: \( P(E \mid I_1, \ldots, I_n) \in \{0, 1\} \) with \( b(I_1, \ldots, I_n) = 1 \) if \( P(e \mid I_1, \ldots, I_n) = 1 \)
Noisy-OR model

Example: suppose we have an OR model and two (true) causes, it follows:

\[
P(e \mid c_1, c_2) = P(i_1 \mid c_1)P(\neg i_2 \mid c_2) + P(i_1 \mid c_1)P(\neg i_2 \mid c_2) + P(i_1 \mid c_1)P(i_2 \mid c_2)
\]

Compare: \( \mathcal{R} = \{ E(x) \leftarrow \alpha(x, k, y), C(k, y) \} \) with \( \alpha(x, k, y) \) and \( C(k, y) \) assumables
Noisy-OR model (2)

\[
P(E(t) \mid C_1(t), C_2(t))
\]
\[
= \frac{P(E(t), C_1(t), C_2(t))}{P(C_1(t), C_2(t))}
\]
\[
= \frac{P((\alpha(t,1,t) \land C_1(t) \land C_2(t)) \lor (\alpha(t,2,t) \land C_1(t) \land C_2(t)))}{P(C_1(t) \land C_2(t))}
\]
\[
= \frac{P(C_1(t) \land C_2(t) \land (\alpha(t,1,t) \lor \alpha(t,2,t)))}{P(C_1(t) \land C_2(t))}
\]
\[
= P(\alpha(t, 1, t) \lor \alpha(t, 2, t))
\]
\[
= P(\alpha(t, 1, t) \land \alpha(t, 2, t)) + P(\alpha(t, 1, t) \land \alpha(f, 2, t)) +
P(\alpha(f, 1, t) \land \alpha(t, 2, t))
\]
\[
= P(\alpha(t, 1, t))P(\alpha(t, 2, t)) + P(\alpha(t, 1, t))P(\alpha(f, 2, t)) +
P(\alpha(f, 1, t))P(\alpha(t, 2, t))
\]
Goal: Probabilistic Logic Learning

Probability

Logic

Learning
Example: Markov Logic Networks

- Specify undirected models (Markov networks) with the use of first-order logic
- An interface layer for AI (Domingos)
- Syntax: weighted first-order formulas
- Semantics: feature templates for Markov networks
- Intuition: soften logical constraints
  - Each formula has a weight
  - Higher weights mean stronger constraints
  - \( P(\text{world}) \exp(\text{sum weights of formulas it satisfies}) \)
Example: Markov Logic Networks(2)

1.5  \( \forall x \text{ MentionOf}(x, \text{Obama}) \Rightarrow \text{Head}(x, "\text{Obama}") \)

0.8  \( \forall x \text{ MentionOf}(x, \text{Obama}) \Rightarrow \text{Head}(x, "\text{President}") \)

100  \( \forall x, y, c \text{ Apposition}(x, y) \land \text{MentionOf}(x, c) \Rightarrow \text{MentionOf}(y, c) \)

Two mention constants: A and B

Example from Poone's NAACL tutorial slides
Example: ProbLog

- Probabilistic Prolog (Problog) at KU Leuven
- Probabilistic programming language (extends YAP Prolog, as opposed to meta-interpreter such as AILog)
- Supports probabilistic inference/learning in Prolog
- Standard clauses: `path(X, Y, A) : ¬X ≠ Y, edge(X, Y), etc.`
- Probabilistic facts: `0.9 :: edge(y, 1, 2)`
- Syntactic sugar:
  `0.3 :: fall(X) ∨ 0.7 :: stay(X) ← manipulate(X)`
- Basically extends standard Prolog queries like `?- q(X).` and `?- mother(a, Y) to ?- prob(q(X)).` and `?- prob(mother(a, Y))`
Research topics

- Efficient inference: given the explanations, it is not so easy to compute the probability, e.g., consider this:

$$P(a \lor b \lor c) = P(a) + P(a \land \neg b) + P(a \land \neg b \land \neg c)$$

- Semantical questions such as dealing with hard constraints. Recall that in abduction we have **nogoods**, `false <- chills`

What does this mean for the probability distribution? And would it possible to add **soft constraints**?

- **Learning** clauses and parameters from data

- Application oriented: many **application fields** are both relational as well as probabilistic