Lecture 5: Logic Programming

2-AIN-108 Computational Logic

Martin Baláž, Martin Homola

Department of Applied Informatics
Faculty of Mathematics, Physics and Informatics
Comenius University in Bratislava

11 Nov 2014

Problem Description

Why classical logics (like propositional or first-order logic) fail to model human reasoning?

Problem Description

Birds usually fly. Penguins are birds. They can not fly (neither birds with a broken wing, ...). Skippy is a bird. Tweety is a penguin. Does Skippy fly? Does Tweety fly?

Problem Description

Birds usually fly. Penguins are birds. They can not fly (neither birds with a broken wing, ...). Skippy is a bird. Tweety is a penguin. Does Skippy fly? Does Tweety fly?

First-order theory T:

$$(\forall x)(\mathit{bird}(x) \land \neg \mathit{penguin}(x) \land \cdots \rightarrow \mathit{fly}(x))$$
$$(\forall x)(\mathit{penguin}(x) \rightarrow \mathit{bird}(x))$$
$$\mathit{bird}(\mathit{Skippy})$$
$$\mathit{penguin}(\mathit{Tweety})$$

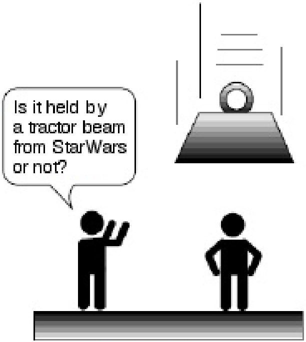
Query:

$$T \models \mathit{fly}(\mathit{Skippy})?$$
$$T \models \mathit{fly}(\mathit{Tweety})?$$

Problem Description

$$T \not\models fly(Skippy)$$
$$T \cup \{\neg penguin(Skippy), \dots\} \models fly(Skippy)$$

Although human knowledge is usually incomplete, we are still able to infer reasonable conclusions.



Solution Proposal

We introduce new type of negation: negation as failure.

A formula $\sim penguin(Skippy)$ is true (resp. $penguin(Skippy)$ is false) if we fail to prove $penguin(Skippy)$.

There is difference between

- having an evidence for the classically negated atom $\neg penguin(Skippy)$
- missing an evidence for the atom $penguin(Skippy)$

In the case of an incomplete information, the classical negation $\neg penguin(Skippy)$ is not inferred but the negation as failure $\sim penguin(Skippy)$ is.

Solution Proposal

Birds usually fly. Penguins are birds. They can not fly (neither birds with a broken wing, ...). Skippy is a bird. Tweety is a penguin. Does Skippy fly? Does Tweety fly?

Logic program P:

$$\begin{aligned} \mathit{fly}(x) &\leftarrow \mathit{bird}(x), \sim \mathit{penguin}(x), \dots \\ \mathit{bird}(x) &\leftarrow \mathit{penguin}(x) \\ \mathit{bird}(\mathit{Skippy}) &\leftarrow \\ \mathit{penguin}(\mathit{Tweety}) &\leftarrow \end{aligned}$$

Query:

$$P \models \mathit{fly}(\mathit{Skippy})?$$
$$P \models \mathit{fly}(\mathit{Tweety})?$$

Outline

1. We define the syntax and semantics of logic programs.
2. We show how backward chaining can be used for query answering in PROLOG.
3. We show how forward chaining can be used for computing stable models in Answer Set Programming.
4. We compare both approaches.

Definition (Literal)

A literal is an atom or an atom preceded by negation $\sim$.

Definition (Clause)

A clause is a disjunction of literals.

Definition (Rule)

A rule is a formula of the form

$$A_0 \leftarrow A_1, \ldots, A_m, \sim A_{m+1}, \ldots, \sim A_n$$

where $0 \leq m \leq n$ and each A_i, $0 \leq i \leq n$, is an atom.

Definition (Program)

A logic program is a set of rules.

Rules

Each rule

$$A_0 \leftarrow A_1, \ldots, A_m, \sim A_{m+1}, \ldots, A_n$$

can be viewed as an implication

$$A_1 \wedge \cdots \wedge A_m \wedge \sim A_{m+1} \wedge \cdots \wedge \sim A_n \rightarrow A_0$$

and equivalently as a clause

$$\sim A_1 \vee \cdots \vee \sim A_m \vee A_{m+1} \vee \cdots \vee A_n \vee A_0$$

A fact is a rule of the form

$$A \leftarrow$$

A constraint is a rule of the form

$$\leftarrow A_1, \ldots, A_m, \sim A_{m+1}, \ldots, \sim A_n$$

Example

Consider the following logic program P:

$$
\begin{aligned}
p(X, Y) &\leftarrow e(X, Y) \\
p(X, Y) &\leftarrow e(X, Z), p(Z, Y) \\
e(a, b) &\leftarrow \\
e(b, c) &\leftarrow
\end{aligned}
$$

and the atom $A = p(a, c)$.

What is the meaning of the logic program P?
What we need to do to check if $P \models A$?

Herbrand Interpretation

Definition (Herbrand Universe)

A term is ground if it does not contain variables.
The Herbrand universe is the set $\mathcal{U}$ of all ground terms.

Definition (Herbrand Base)

An atom is ground if it does not contain variables.
The Herbrand base is the set $\mathcal{B}$ of all ground atoms.

Definition (Herbrand Interpretation)

A Herbrand interpretation is an interpretation $\mathcal{I} = (\mathcal{U}, I)$ such that

$$f^I = (t_1, \ldots, t_n) \mapsto f(t_1, \ldots, t_n)$$

for each function symbol f with arity n.

Theorem

A logic program is satisfiable iff it has a Herbrand model.

Sketch of proof.

Each Herbrand model is a model, i.e. if a logic program has a Herbrand model, it has a model.
If $\mathcal{I} = (D, I)$ is a model of P then a Herbrand interpretation $\mathcal{J} = (\mathcal{U}, J)$ such that

$$J(p) = \{(t_1, \ldots, t_n) \mid I \models p(t_1, \ldots, t_n)\}$$

is a Herbrand model of P. □

Properties

The previous theorem holds only for clauses, it does not hold for arbitrary closed formulas.

Let S be $\{p(a), (\exists X)\neg p(X)\}$. The Herbrand universe is $\mathcal{U} = \{a\}$ and the Herbrand base is $\mathcal{B} = \{p(a)\}$. We have two Herbrand interpretations, $(\{a\}, I_1)$, $p^{I_1} = \emptyset$ (i.e. $p(a)$ is false), and $(\{a\}, I_2)$, $p^{I_2} = \{(a)\}$ (i.e. $p(a)$ is true). In both cases, S is not satisfied.

But if we take the domain $D = \{0, 1\}$ and the interpretation function I_3 with $a^{I_3} = 0$, $p^{I_3} = \{(0)\}$, then (D, I_3) is a model of S.

Definite Logic Program

Definition (Definite Rule)

A definite rule is a rule of the form

$$A_0 \leftarrow A_1, \ldots, A_n$$

where $0 \leq n$ and each A_i, $0 \leq i \leq n$, is an atom.

Definition (Definite Logic Program)

A logic program is definite if it contains only definite rules.

The Least Herbrand Model

Lemma

Let P be a definite logic program and $\mathcal{M}$ be a non-empty set of Herbrand models of P. Then $\bigcap_{M \in \mathcal{M}} M$ is a Herbrand model of P.

Theorem

Every definite logic program P has the least Herbrand model (denoted M_P).

Proof.

The set of all Herbrand models is non-empty, because the Herbrand base $\mathcal{B}$ is a model of P. The intersection of all Herbrand models is the least Herbrand model of P. $\square$

The Least Herbrand Model

Theorem

Let P be a definite logic program. Then $M_P = \{A \in \mathcal{B}_P \mid P \models A\}$.

Proof.

$P \models A$ iff $P \cup \{\sim A\}$ is unsatisfiable iff $P \cup \{\sim A\}$ has no Herbrand models iff $\sim A$ is false w.r.t. all Herbrand models of P iff A is true w.r.t. all Herbrand models of P iff $A \in M_P$. $\square$

Definition (Immediate Consequence Operator)

Let P be a definite logic program. An immediate consequence operator T_P is defined as follows:

$$T_P(I) = \{A \in \mathcal{B}_P \mid A \leftarrow A_1, \dots, A_n \in Ground(P), \\ \{A_1, \dots, A_m\} \subseteq I\}$$

The iteration $T_P \uparrow n$ is defined as follows:

$$\begin{aligned} T_P \uparrow 0 &= \emptyset \\ T_P \uparrow (n+1) &= T_P(T_P \uparrow n) \\ T_P \uparrow \omega &= \bigcup_{n<\omega} T_P \uparrow n \end{aligned}$$

Theorem

Let M_P be the least model of P. Then $M_P = T_P \uparrow \omega$.

Definition (Normal Rule)

A normal rule is a rule of the form

$$A \leftarrow L_1, \ldots, L_n$$

where $0 \leq n$, A is an atom, and each L_i, $1 \leq i \leq n$, is a literal.

Definition (Normal Logic Program)

A logic program is normal if it contains only normal rules.

Reasoning with Negation

$$\begin{aligned} student(joe) &\leftarrow \\ student(bill) &\leftarrow \end{aligned}$$

$$P \models student(jim)?$$

$$P \models \sim student(jim)?$$

$$student(x) \leftrightarrow x = joe \lor x = bill$$

First step:

$$p(x_1, \ldots, x_m) \leftarrow x_1 = t_1 \wedge \cdots \wedge x_m = t_m \wedge L_1 \wedge \cdots \wedge L_n$$

where $x_1, \ldots, x_m$ are variables not occuring in $L_1, \ldots, L_n$ and $p(t_1, \ldots, t_m) \leftarrow L_1, \ldots, L_n$ is a normal rule.

Second step:

$$p(x_1, \ldots, x_m) \leftrightarrow E_1 \vee \cdots \vee E_k$$

where each E_i has the form $x_1 = t_1 \wedge \cdots \wedge x_m = t_m \wedge L_1 \wedge \cdots \wedge L_n$, $E_1, \ldots E_k$ are all transformed rules from the first step with the predicate symbol p in the head, and $x_1, \ldots, x_m$ are new variables.