VNU-HUS MAT1206E/3508: Introduction to AI

Limitations of Logic

In-class Discussion

Hoàng Anh Đức

Bộ môn Tin học, Khoa Toán-Cơ-Tin học
Đại học KHTN, ĐHQG Hà Nội
hoanganhduc@hus.edu.vn

Contents

The Search Space Problem

The Search Space Problem

- In the search for a proof, depending on the calculus, potentially *there are (infinitely) many ways to apply inference rules at each step*
- After several inference steps, *the number of possible sequences can be enormous*

Because of the search space problem, *automated provers* today can *only prove relatively simple theorems in special domains with few axioms*

The Search Space Problem

Quiz

In resolution proof search, each pair of clauses with complementary literals gives one possible *application of the resolution rule*. Suppose a prover has *four* possible applications at each step. If the prover makes *five* resolution steps, how many possible sequences of inference steps can it explore?

(a) $4 + 5 = 9$

(b) $4 \cdot 5 = 20$

(c) $4^5 = 1024$

The Search Space Problem

	Automated Provers	***Human Experts***
Number of inferences performed per second	*20000*	*1*
Solve difficult problems	*slow*	*fast*

Reasons

- Humans use *intuitive calculi* that work on a higher level
  - These intuitive calculi often *carry out many of the simple inferences of an automated prover in one step*
- Humans work with *lemmas*
  - We already know the lemmas are true and *do not need to re-prove* them each time
- Humans use *intuitive meta knowledge* (informal!)
  - *Intuition* is an important advantage of humans. Without it, we could not solve any difficult problems
- Humans use *heuristics*
  - Heuristics often *shorten the way to the goal*; rarely, they can make the search longer

The Search Space Problem

Problems

- Often humans are *unable to formulate intuitive meta-knowledge verbally*!
- Humans *learn heuristics by experience*

Solution 1

Try to "copy nature": learn heuristics using machine learning techniques

Solution 2

- Computer algebra systems such as Mathematica, Maple, or Maxima can automatically carry out difficult symbolic mathematical manipulations, but proof search is left fully to the human
- Interactive theorem provers, for example Isabelle (`https://isabelle.in.tum.de/`), check and support user-guided proof search

Decidability and Incompleteness

- *Decidability* refers to whether there exists an algorithm that can determine the truth or falsity of any statement in a given logical system.
- *Incompleteness* refers to the fact that in any sufficiently powerful logical system, there are statements that are true but cannot be proven within the system.

Examples?

- The *Halting Problem* is a classic example of an *undecidable problem*.
- In first-order logic, if a given formula is valid, there is a proof of it (Gödel's Completeness Theorem). However, if the formula is not valid, it may happen that the prover never halts (undecidability). On the other hand, propositional logic is decidable.
- Gödel's *Incompleteness Theorems* show that in any consistent formal system that is capable of expressing basic arithmetic, there are true statements that cannot be proven within the system.

Decidability and Incompleteness

Decidable means that one finite procedure answers every input. PL1 has only a one-sided guarantee.

Quiz

Which statement is correct?

(a) Complete calculi make PL1 decidable.

(b) Truth tables decide propositional logic; valid PL1 formulas are eventually proved.

(c) Non-valid PL1 formulas are always refuted.

Decidability and Incompleteness

Quiz

A truth-table method decides propositional formulas, but in the worst case it may need 2^n rows for n variables. What does this show?

(a) Propositional logic is undecidable.

(b) Decidable does not necessarily mean practical.

(c) Truth tables are incomplete.

Decidability and Incompleteness

Note

The deeper background is that the more expressive a formal language is, the easier it becomes to write statements that talk about themselves or lead to paradox-like situations.

Example 1 ("Too powerful language")

- A warning example is *naive set comprehension*: if every condition is allowed to define a set, then $M = \{x \mid x \notin x\}$ leads to Russell's paradox.
- The barber story has the same self-reference pattern: a barber shaves exactly those people who do not shave themselves.

Example: The Flying Penguin

Knowledge Base (KB)

(1) Tweety is a penguin
(2) Penguins are birds
(3) Birds can fly

(1) *penguin*(*Tweety*)
(2) $\forall x\,(\textit{penguin}(x) \Rightarrow \textit{bird}(x))$
(3) $\forall x\,(\textit{bird}(x) \Rightarrow \textit{fly}(x))$

Query (Q)

Can *Tweety fly*? ($Q = \textit{fly}(\textit{Tweety})$)

Note that $KB \vdash Q$, i.e., Q can be proven from KB.

Example: The Flying Penguin

Quiz

How can $KB \vdash Q$ be shown for $Q = \textit{fly}(\textit{Tweety})$?

(a) Use the two rules with $x = \textit{Tweety}$.

(b) Use the extra rule that penguins cannot fly.

(c) It cannot be derived because Tweety is a penguin.

Example: The Flying Penguin

However, in reality, *penguins cannot fly*. That is, we should prevent the derivation of Q from KB

First Attempt

To mimics the reality, add $\forall x\,(\textit{penguin}(x) \Rightarrow \neg \textit{fly}(x))$ to KB.

Quiz

After adding $\forall x(\textit{penguin}(x) \Rightarrow \neg \textit{fly}(x))$, what can be derived?

(a) Only $\neg Q$ (b) Only Q (c) Both Q and $\neg Q$

Example: The Flying Penguin

Second Attempt

- Add “Penguins cannot fly” to the original KB
- Replace the non-realistic rule “Birds can fly” by “Birds except penguins can fly”.

Knowledge Base (KB_2)

(1) Tweety is a penguin	(1) *penguin*(*Tweety*)
(2) Penguins are birds	(2) $\forall x\,(\textit{penguin}(x) \Rightarrow \textit{bird}(x))$
(3) Birds except penguins can fly	(3) $\forall x\,(\textit{bird}(x) \land \neg\textit{penguin}(x) \Rightarrow \textit{fly}(x))$
(4) Penguins cannot fly	(4) $\forall x\,(\textit{penguin}(x) \Rightarrow \neg\textit{fly}(x))$

Quiz

Why can $Q = \textit{fly}(\textit{Tweety})$ *not* be derived from KB_2?

(a) No bird fact. (b) Missing $\neg\textit{penguin}(\textit{Tweety})$. (c) No fly rule.

Example: The Flying Penguin

But ...

- Whenever we want to add a *new bird*, we have to *specify whether it can fly or not*
- This means we have to *explicitly specify negative information*: the newly added bird is *not a penguin* (because of the rule "Birds except penguins can fly"); otherwise, we have *no conclusion* about whether the new bird can fly or not
- For the construction of a knowledge base with all 9800 or so types of birds worldwide, this becomes a *significant challenge*.

More generally, classical PL1 does not treat an unstated fact as false. If a rule needs $\neg P(a)$, the knowledge base must contain enough information to derive $\neg P(a)$ (here, the flying rule needs $\neg\textit{penguin}(\textit{abraxas})$, so KB_4 adds $\textit{raven}(x) \Rightarrow \neg\textit{penguin}(x)$ to combine with $\textit{raven}(\textit{abraxas})$). In large domains, many such negative facts or rules may have to be stated explicitly.

Example: The Flying Penguin

The *frame problem* asks how a knowledge base can avoid listing, for every action, all facts that stay unchanged unless the action affects them.

Example 2 (An example of the frame problem)

- A blue house is painted red, then afterwards it is red
- However, with the knowledge base

$$\mathit{color}(\mathit{house}, \mathit{blue})$$
$$\mathit{paint}(\mathit{house}, \mathit{red})$$
$$\mathit{paint}(x, y) \Rightarrow \mathit{color}(x, y)$$

  one can derive $\mathit{color}(\mathit{house}, \mathit{red})$
- The old fact $\mathit{color}(\mathit{house}, \mathit{blue})$ is still in the knowledge base, so the house is represented as both blue and red after painting.

Example: The Flying Penguin

Some Resolving Ideas

- For the *frame problem*, a standard treatment is to represent time or situations explicitly, for example with *situation calculus*.
  - Plain idea: separate what is true before an action from what is true after it.
- For the default-assumption problem in examples such as Tweety, *non-monotonic logics* have been developed.
  - Previously drawn conclusions or *default assumptions* can be *withdrawn when new information is added*.
  - Caveat: defaults are useful, but choosing rules that behave correctly in all cases is difficult.
- Another approach for modeling uncertainty is *probability theory*.

Modelling Uncertainty

- *Two-valued logic* can and should only *model circumstances in which there is true, false, and no other truth values*.
- For many tasks in everyday reasoning, *two-valued logic is therefore not expressive enough*.
  - For example, the rule *bird*$(x) \Rightarrow$ *fly*(x) is *true for almost all birds, but for some it is false*.
- As we already mentioned, to formulate uncertainty, we can use *probability theory*.
  - Informal shorthand: $P($*bird*$(x) \Rightarrow$ *fly*$(x)) = 0.99$ means *"about 99% of birds can fly"*.
  - Later, we will see that here it is better to work with *conditional probabilities* such as $P($*fly*$|$*bird*$) = 0.99$. With the help of *Bayesian networks*, complex applications with many variables can also be modelled.
  - *Fuzzy logic* is one model for *"The weather is nice"*, where strict true/false values are often unnatural.
  - The variable *weather_is_nice* is *continuous with values in $[0, 1]$*. *weather_is_nice* $= 0.7$ then means *"The weather is fairly nice"*.

Modelling Uncertainty

Quiz

For $P(\mathit{fly}|\mathit{bird}) = 0.99$ and *weather_is_nice* $= 0.7$, which interpretation is correct?

(a) Both numbers are probabilities of true/false statements.

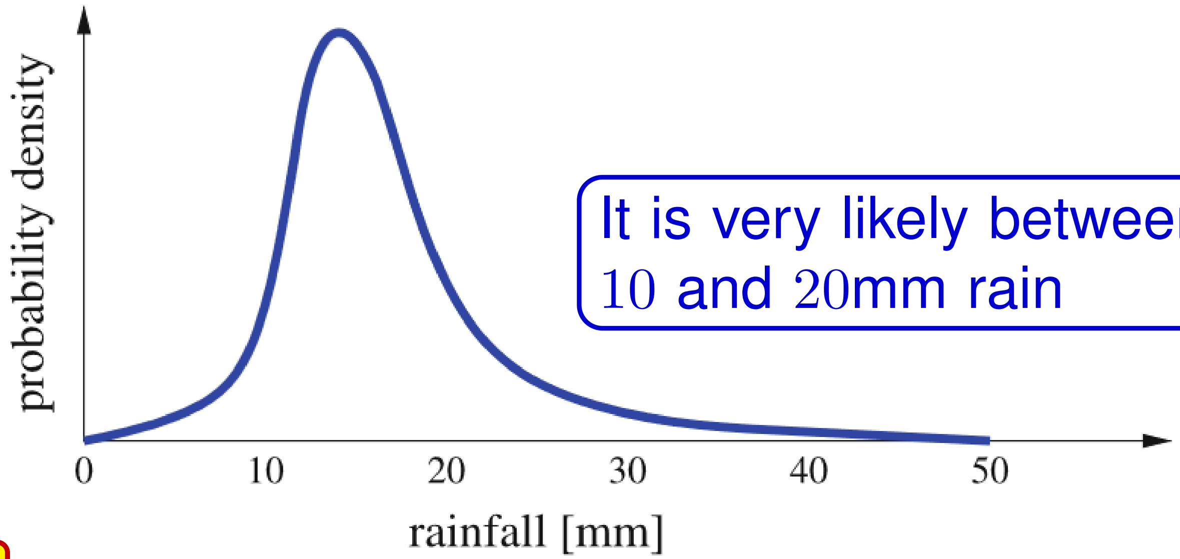
(b) 0.99 is a probability; 0.7 is a fuzzy degree for "the weather is nice".

(c) 0.99 is a fuzzy degree; 0.7 is a probability.

Modelling Uncertainty

- Probability theory can also model *continuous variables* using *probability densities*.

"There is a high probability that there will be some rain"



It is very likely between 10 and 20mm rain

Note

This very *general and even visualizable representation* of both types of uncertainty we have discussed, together with *inductive statistics* and the theory of *Bayesian networks*, makes it possible, in principle, *to answer arbitrary probabilistic queries*.

Modelling Uncertainty

Comparison of different formalisms for the modelling of uncertain knowledge

Formalism	Values used	Meaning
Propositional logic	2	truth values: true/false
Fuzzy logic	∞	degrees of truth
Discrete probabilistic logic	n	discrete probability values
Continuous probabilistic logic	∞	*probability values in* $[0, 1]$

- These formalisms are not directly comparable to predicate logic; here we compare simple propositional-style uncertainty formalisms.