Automated Reasoning

Jacques Fleuriot

September 14, 2013

Representation

Jacques Fleuriot

Representing Knowledge

We are faced with several choices when formalising a theory:

- Which **type of logic** to use?
  - Propositional Logic;
  - First-Order Logic;
  - Others (such as Higher-Order Logic, which we will cover)
- Do we need axioms?
- How do we represent the concepts of our domain?

Logic and Notation

- Early slides use mixture of single-sorted FOL and multi-sorted FOL formulas, though also can be read as HOL formulas.
- Later will introduce HOL and contrast it with multi-sorted FOL
- Multi-sorted FOL primer
  - Sorts (types): *bool*, *int*, *real*, $\alpha \times \beta$, α*set*
  - Still **terms** (individuals) and **formulas** are distinct syntactic categories
  - Use $\top$ and $\bot$ both as formulas and terms of *bool* sort. Intent will be clear from context
  - Functions have argument and result sorts: $f : (\alpha, \beta)\gamma$
  - Relations have argument sorts: $R : (\alpha, \beta)$

Axioms

Consider the natural numbers $\mathbb{N} = \{0, 1, 2, \ldots\}$. How do we prove facts about them? For example: how do we prove that every natural number greater than 1 has a prime divisor?

Axiomatically

- We take natural numbers as *primitive*, and assume unproven axioms about them. For instance, we assume the Peano axioms:

$$\begin{aligned}
&\forall n.\ n + 0 = n. \\
&\forall m\, n.\ (m + S(n)) = S(m + n) \\
&\ldots
\end{aligned}$$

- Everything we want to prove about natural numbers are proven from the axioms.
- But how do we know that our axioms are **adequate**? Are they complete?
- How do we know that our axioms are **consistent**?

Axioms and Definition (II)

Conservatively

- We *define* the natural numbers *in terms* of other objects. For instance, we identify the natural numbers with Von Neumann ordinals: $0 = \emptyset$, $1 = \{\emptyset\}$, $2 = \{\emptyset, \{\emptyset\}\}$, ... The theory of natural numbers is then the theory of Von Neumann ordinals.
- But how do we find suitable definitions?

We can mix this with the axiomatic approach: we define natural numbers in terms of Von Neumann ordinals and *then* prove the Peano Axioms on this interpretation.
This approach guarantees **relative consistency**: if the theory of Von Neumann ordinals is consistent, so is the theory of natural numbers.

Representation Examples (Integers)

Starting from the natural numbers $\mathbb{N} = \{0, 1, 2, \ldots\}$, we can define:

- each integer $\mathbb{Z} = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$ as an **equivalence class** of pairs of natural numbers under the relation $(a, b) \sim (c, d) \longleftrightarrow a + d = b + c$;
- For example, -2 is represented by the equivalence class $[(0, 2)] = [(1, 3)] = [(100, 102)] = \ldots$.
- we define the sum and product of two integers as

$$\begin{aligned}
[(a, b)] + [(c, d)] &= [(a + c, b + d)] \\
[(a, b)] \times [(c, d)] &= [(ac + bd, ad + bc)]\,;
\end{aligned}$$

- we define the set of **negative** integers as the set $\{[(a, b)] \mid b > a\}$.
- Exercise: show that the product of negative integers is non-negative.

Other Representation Examples

- The rationals $\mathbb{Q}$ can be defined as pairs of integers. Reasoning about the rationals therefore reduces to reasoning about the integers.
- The reals $\mathbb{R}$ can be defined as sets of rationals. Reasoning about the reals therefore reduces to reasoning about the rationals.
- The complex numbers $\mathbb{C}$ can be defined as pairs of reals. Reasoning about the complex numbers therefore reduces to reasoning about the reals.
- In this way, we have **relative consistency**.
  - If the theory of natural numbers is consistent, so is the theory of complex numbers.

Functional Representation

- When defining concepts in our theory, we often have a choice between using functions and predicates.
- For example, suppose we represent division of real numbers ($/$) by a function $div : (real, real)real$.
  - We define $div(x, y)$ when $y \neq 0$ in normal way
  - What about division-by-zero? What is the value of $div(x, 0)$?
  - In first-order logic, functions are assumed to be **total**, so we have to pick a value!
  - We could *choose* a convenient element: say 0. That way:

$$0 \leq x \rightarrow 0 \leq 1/x.$$

Predicate Representation

Q) Can we represent division of real numbers ($/$) by a relation $Div : (real, real, real)$ such that $Div(x, y, z)$ is

- $x/y = z$ when $y \neq 0$, and
- $\perp$ when $y = 0$?

A) Yes: $Div(x, y, z) \;\equiv\; x = y * z \;\wedge\; \forall w.\, x = y * w \longrightarrow z = w$
That is, z is that *unique* value such that $x = y * z$.

But now formulas are more complicated.

$$x, y \neq 0 \longrightarrow \frac{1}{((x/y)\,/x)} = y$$

becomes

$$Div(x, y, u) \wedge Div(u, x, v) \wedge Div(1, v, w) \wedge x, y \neq 0 \longrightarrow w = y$$

Functional Representation

Can we represent the concept of *square roots* with a function $\sqrt{} : (\mathit{real})\mathit{real}$?

- All positive real numbers have *two* square roots, and yet a function maps points to *single* values.
- We can pick one of the values arbitrarily: say, the *positive* (*principal*) square root.
- **Or** we can have the function map every real to a *set* $\sqrt{} : (\mathit{real})\mathit{real\ set}$:

$$\sqrt{x} \equiv \left\{ y \mid x = y^2 \right\}.$$

- But now we have two kinds of object: reals and sets of reals, and we cannot conveniently express:

$$(\sqrt{x})^2 = x$$

- Our representation of reals is no longer **self-contained**.

Predicate Representation

Q) Can we represent the concept of *square roots* with a relation $Sqrt : (real, real)$?

A) Yes. E.g. $Sqrt(x, y) \equiv x = y^2$.

Again drawback of formulas being more complicated

Functions, Predicates and Sets

Any function $f : (\alpha)\beta$ can be represented as a relation $R : (\alpha, \beta)$ or a set $S : (\alpha \times \beta)set$ by defining:

$$\begin{aligned} R(x, y) &\equiv f(x) = y \\ S &\equiv \{(x, y) \mid f(x) = y\}. \end{aligned}$$

Any predicate P can be represented by a function f or a set S by defining:

$$\begin{aligned} f(x) &\equiv \begin{cases} True & : \quad P(x) \\ False & : \quad \text{otherwise} \end{cases} \\ S &\equiv \{x \mid P(x)\}. \end{aligned}$$

Any set S can be represented by a function f or a predicate P by defining:

$$\begin{aligned} f(x) &\equiv \begin{cases} True & : \quad x \in S \\ False & : \quad \text{otherwise} \end{cases} \\ P(x) &\equiv x \in S \end{aligned}$$

Set Theory

In **pure** (without axioms) single-sorted **FOL**, we **cannot directly represent** the statement:

> *there is a function that is larger on all arguments than the log function.*

To formalise it, we would need to quantify over functions:

$$\exists f. \forall x. f(x) > \log x.$$

Likewise we cannot quantify over predicates.

Solutions in FOL:

- Represent all functions and predicates by **sets**, and quantify over these. This is the approach of first-order set theories such as *ZF*.
- Introduce sorts for predicates and functions. Not so elegant now having 2 kinds of each.

Higher-Order Logic (HOL)

Alternatively...

In HOL, we represent sets and predicates by **functions**, often denoted by **lambda abstractions**.

Definition (Lambda Abstraction)

Lambda abstractions are **terms** which denote functions directly by the rules which define them. E.g. the square function is denoted by $\lambda x.\, x * x$.

We can use lambda abstractions exactly as we use ordinary function symbols. E.g. $(\lambda x.\, x * x)\, 3 = 9$.

Higher-order Functions

We can define functions which map from and to other functions.

Example

The K-combinator maps some x to a function which sends any y to x.

$$\lambda x.\ \lambda y.\ x.$$

Example

The composition function maps two functions to their composition:

$$\lambda f.\ \lambda g.\ \lambda x.\ f\,(g\,x).$$

Representation of Logic in HOL I

- Types *bool*, *ind* (individuals) and $\alpha \Rightarrow \beta$ primitive. All others defined from these.
- Start with equality function $= : \alpha \Rightarrow \alpha \Rightarrow bool$. All other functions defined using this, lambda abstraction and application.
- Distinction between formulas and terms is dispensed with: formulas are just terms of type *bool*.
- Definition of product type

$$\begin{array}{lcl}
\alpha \times \beta & \doteq & (\alpha \Rightarrow \beta \Rightarrow \gamma) \Rightarrow \gamma \\
(x, y) & \doteq & \lambda f.\, f\, x\, y. \\
\pi_1\ p & \doteq & p(\lambda xy.\, x) \\
\pi_2\ p & \doteq & p(\lambda xy.\, y)
\end{array}$$

Representation of Logic in HOL II

- Conjunction as pairs:

$$x \wedge y \equiv (x, y) = (\mathit{True}, \mathit{True}).$$

- Universal quantification as function equality:

$$\forall x.\ \phi \equiv (\lambda x.\ \phi) = (\lambda x. \mathit{True}).$$

- Predicates and sets can be represented by functions.
- Therefore, we can **quantify over functions, predicates and sets**.