A New Coq Formalisation of Classical First-Order Logic with Proofs of the Soundness and Completeness Theorems

Kijeong Lim

Chonnam National University

The Fourth Korea Logic Day
January 14, 2025

Table of Contents

Introduction

The Motivation.

- It has been a common practice in software verification to rely on proof assistants, such as Coq and Isabelle, to obtain formal proofs of program correctness. In these systems, programs are abstracted as mathematical entities, and their behaviour is denoted by certain mathematical properties.
- To reason about the behaviour of a program effectively using a proof assistant, there must be a well-developed theory about the mathematical properties being verified. For example, Software Foundations Lab formalised a theory of ordinal numbers, which is a purely mathematical concept, to state and/or prove the termination of a given program.
- However, most proof assistants do not adopt the usual first-order language used by working mathematicians. Instead, they take dependent type theories as their foundation. Therefore, to facilitate the translation of mathematical theories into a proof assistant, an intermediate language is deemed necessary.

Introduction

The Advantages of Embedding First-Order Logic into "Coq".

- **CIC** serves as a very rich metalanguage. In contrast, the Fitch system has poor expressiveness, which limits its ability to state and/or prove meta-theorems. For example, it cannot directly state whether an object theory $\mathcal{T}$ of arithmetic admits the ω-rule, whereas Coq can.

$$\frac{(\forall n \in \mathbb{N})(\mathcal{T} \vdash P(\bar{n}))}{\mathcal{T} \vdash \dot{\forall} x\, P(x)}\ \omega\text{-rule}$$

- One can leverage various libraries of the Coq community when formalising an embedded first-order theory. For instance, if a first-order theory $\mathcal{T}$ is complete—that is, $\mathcal{T}$ proves either an arbitrary proposition or its negation—the non-existence of a model for the negation of a proposition φ implies that φ is a theorem of $\mathcal{T}$, where the non-existence may be shown with the libraries in Coq.
- Furthermore, any sentence φ proved in an embedded first-order theory $\mathcal{T}$ can be lifted to its corresponding theorem, which can be directly used in Coq. If $\mathfrak{M}$ is a model of $\mathcal{T}$, then it immediately becomes a theorem in Coq that φ holds for $\mathfrak{M}$.

Introduction

Formalisation Overview.

1. **Syntax.** How to use symbols to define logical expressions.
2. **Semantics.** What is the meaning of a logical expression.
3. **Deduction System.** What formulae are provable.
4. **Meta-theory.** The theory on first-order logic.

Formalisation (Syntax)

A ***first-order language*** L is represented as a record with the following fields:

- a **Set** $\mathcal{F}$ of ***function symbols***;
- a **Set** $\mathcal{C}$ of ***constant symbols***;
- a **Set** $\mathcal{R}$ of ***relation symbols***;
- a table mapping each $f \in \mathcal{F}$ to its arity $n_f \in \mathbb{N}_{>0}$; and
- a table mapping each $R \in \mathcal{R}$ to its arity $n_R \in \mathbb{N}_{>0}$.

Formalisation (Syntax)

- An ***individual variable*** x is of the form v_i for a natural number i—i.e.,

$$x \ ::= \ v_i \quad \text{for} \quad i \in \mathbb{N}.$$

- An L***-term*** t is defined inductively by

$$t \ ::= \ x \mid f\,\vec{t} \mid c$$

  where $f \in \mathcal{F}$, $c \in \mathcal{C}$, and $\vec{t}$ is a vector of L-terms.
- An L***-formula*** φ is defined inductively by

$$\varphi \ ::= \ R\,\vec{t} \mid t_1 \doteq t_2 \mid \dot{\neg}\,\varphi_1 \mid \varphi_1 \mathrel{\dot{\to}} \varphi_2 \mid \dot{\forall} x\,\varphi_1$$

  where $R \in \mathcal{R}$.

Notations. For a first-order language L,

- the type of L-terms is denoted by $\mathtt{trm}\,L$;
- the type of L-formulae is denoted by $\mathtt{frm}\,L$;
- an L-formula may be denoted by φ, ψ, θ, p, q, r, A, B, C, or b; and
- a set of L-formulae may be denoted by Γ, Δ, or X.

Formalisation (Syntax)

- A ***simultaneous substitution*** $[\sigma]X$ is defined on a syntactic object X over L, where $\sigma : \mathbb{N} \to \mathtt{trm}\, L$ is a map that replaces each free occurrence of the individual variable v_i with the L-term $\sigma(i)$ in X for every $i \in \mathbb{N}$.
- $\iota := i \mapsto v_i.\ t/x; \sigma := i \mapsto \textbf{if}\ x = v_i\ \textbf{then}\ t\ \textbf{else}\ \sigma(i).\ t/x := t/x; \iota.$

Formalisation (Syntax)

- A ***simultaneous substitution*** $[\sigma]X$ is defined on a syntactic object X over L, where $\sigma : \mathbb{N} \to \mathtt{trm}\, L$ is a map that replaces each free occurrence of the individual variable v_i with the L-term $\sigma(i)$ in X for every $i \in \mathbb{N}$.
- $\iota := i \mapsto v_i.\ t/x; \sigma := i \mapsto \textbf{if } x = v_i \textbf{ then } t \textbf{ else } \sigma(i).\ t/x := t/x; \iota.$
- $[\sigma](\dot{\forall} x\, \varphi_1) := \textbf{let } x' := \chi(\sigma, \dot{\forall} x\, \varphi_1) \textbf{ in } \dot{\forall} x'\, ([x'/x; \sigma]\varphi_1)$, where

$$\chi(\sigma, \varphi) := \max\{\max(\mathrm{FV}(\sigma(i))) \mid v_i \in \mathrm{FV}(\varphi)\} + 1.$$

Formalisation (Syntax)

- A ***simultaneous substitution*** $[\sigma]X$ is defined on a syntactic object X over L, where $\sigma : \mathbb{N} \to \mathtt{trm}\, L$ is a map that replaces each free occurrence of the individual variable v_i with the L-term $\sigma(i)$ in X for every $i \in \mathbb{N}$.
- $\iota := i \mapsto v_i.\ t/x; \sigma := i \mapsto \textbf{if } x = v_i \textbf{ then } t \textbf{ else } \sigma(i).\ t/x := t/x; \iota.$
- $[\sigma](\dot{\forall} x\, \varphi_1) := \textbf{let } x' := \chi(\sigma, \dot{\forall} x\, \varphi_1) \textbf{ in } \dot{\forall} x'\, ([x'/x; \sigma]\varphi_1)$, where

$$\chi(\sigma, \varphi) := \max\{\max(\mathrm{FV}(\sigma(i))) \mid v_i \in \mathrm{FV}(\varphi)\} + 1.$$

- α***-equivalence*** is defined inductively. The introduction rule for $\dot{\forall}$ is

$$\frac{[x'/x_1]\varphi_1 \equiv_\alpha [x'/x_2]\varphi_2}{\dot{\forall} x_1\, \varphi_1 \equiv_\alpha \dot{\forall} x_2\, \varphi_2}\ \equiv_\alpha\text{-}\dot{\forall}$$

  provided by $x' \notin \mathrm{FV}(\dot{\forall} x_1\, \varphi_1) \wedge x' \notin \mathrm{FV}(\dot{\forall} x_2\, \varphi_2)$.

Formalisation (Syntax)

- A ***simultaneous substitution*** $[\sigma]X$ is defined on a syntactic object X over L, where $\sigma : \mathbb{N} \to \mathtt{trm}\, L$ is a map that replaces each free occurrence of the individual variable v_i with the L-term $\sigma(i)$ in X for every $i \in \mathbb{N}$.
- $\iota := i \mapsto v_i.\ t/x; \sigma := i \mapsto \textbf{if } x = v_i \textbf{ then } t \textbf{ else } \sigma(i).\ t/x := t/x; \iota.$
- $[\sigma](\dot{\forall} x\, \varphi_1) := \textbf{let } x' := \chi(\sigma, \dot{\forall} x\, \varphi_1) \textbf{ in } \dot{\forall} x'\, ([x'/x; \sigma]\varphi_1)$, where

$$\chi(\sigma, \varphi) := \max\{\max(\mathrm{FV}(\sigma(i))) \mid v_i \in \mathrm{FV}(\varphi)\} + 1.$$

- α***-equivalence*** is defined inductively. The introduction rule for $\dot{\forall}$ is

$$\frac{[x'/x_1]\varphi_1 \equiv_\alpha [x'/x_2]\varphi_2}{\dot{\forall} x_1\, \varphi_1 \equiv_\alpha \dot{\forall} x_2\, \varphi_2}\ \equiv_\alpha\text{-}\dot{\forall}$$

  provided by $x' \notin \mathrm{FV}(\dot{\forall} x_1\, \varphi_1) \wedge x' \notin \mathrm{FV}(\dot{\forall} x_2\, \varphi_2)$.
- A ***single substitution*** $\varphi[x := t]$ is also defined: $\varphi[x := t] \equiv_\alpha [t/x]\varphi$;

  whenever $x \neq x' \wedge x' \in \mathrm{FV}(t)$, $(\dot{\forall} x'\, \varphi)[x := t] = \dot{\forall} x''\, (\varphi[x' := x''][x := t])$

  for some $x'' \notin \{x\} \cup \mathrm{FV}(t) \cup \mathrm{FV}(\varphi)$; and $(\dot{\forall} x\, \varphi)[x := t] = (\dot{\forall} x\, \varphi)$.

Formalisation (Semantics)

Definition. An *L-**structure*** $\mathfrak{A}$ is a record consisting of the following fields:

- a setoid $(A, \sim_A)$,
- ***interpretations*** $f^{\mathfrak{A}} : A^{n_f} \to A$ for each $f \in \mathcal{F}$,
- ***interpretations*** $c^{\mathfrak{A}} \in A$ for each $c \in \mathcal{C}$, and
- ***interpretations*** $R^{\mathfrak{A}} : A^{n_R} \to \mathbf{Prop}$ for each $R \in \mathcal{R}$,

such that $\sim_A$ is compatible with all $f^{\mathfrak{A}}$ and $R^{\mathfrak{A}}$—i.e.,

$$\frac{a_1 \sim_A a'_1 \quad \cdots \quad a_{n_f} \sim_A a'_{n_f}}{f^{\mathfrak{A}}(a_1, \cdots, a_{n_f}) \sim_A f^{\mathfrak{A}}(a'_1, \cdots, a'_{n_f})} \qquad \frac{a_1 \sim_A a'_1 \quad \cdots \quad a_{n_R} \sim_A a'_{n_R}}{R^{\mathfrak{A}}(a_1, \cdots, a_{n_R}) \leftrightarrow R^{\mathfrak{A}}(a'_1, \cdots, a'_{n_R})}$$

—and A is nonempty.
Denote the type A by $|\mathfrak{A}|$, which is called the ***domain of discourse*** of $\mathfrak{A}$.

Formalisation (Semantics)

Tarski's definition of truth. Let $\mathfrak{A}$ be an L-structure and let $\rho : \mathbb{N} \to |\mathfrak{A}|$. The map $t \mapsto [\![t]\!]^{\mathfrak{A}}_{\rho} : \mathtt{trm}\, L \to |\mathfrak{A}|$ ***interprets*** L-terms, given by

- $[\![v_i]\!]^{\mathfrak{A}}_{\rho} := \rho(i)$,
- $[\![f\ \vec{t}]\!]^{\mathfrak{A}}_{\rho} := f^{\mathfrak{A}}\ [\![\vec{t}]\!]^{\mathfrak{A}}_{\rho}$,
- $[\![c]\!]^{\mathfrak{A}}_{\rho} := c^{\mathfrak{A}}$.

The map $\varphi \mapsto [\![\varphi]\!]^{\mathfrak{A}}_{\rho} : \mathtt{frm}\, L \to \mathbf{Prop}$ ***interprets*** L-formulae, given by

- $[\![R\ \vec{t}]\!]^{\mathfrak{A}}_{\rho} := R^{\mathfrak{A}}\ [\![\vec{t}]\!]^{\mathfrak{A}}_{\rho}$,
- $[\![t_1 \doteq t_2]\!]^{\mathfrak{A}}_{\rho} := [\![t_1]\!]^{\mathfrak{A}}_{\rho} \sim_{|\mathfrak{A}|} [\![t_2]\!]^{\mathfrak{A}}_{\rho}$,
- $[\![\dot{\neg}\, \varphi_1]\!]^{\mathfrak{A}}_{\rho} := \neg\, [\![\varphi_1]\!]^{\mathfrak{A}}_{\rho}$,
- $[\![\varphi_1 \dot{\to} \varphi_2]\!]^{\mathfrak{A}}_{\rho} := [\![\varphi_1]\!]^{\mathfrak{A}}_{\rho} \to [\![\varphi_2]\!]^{\mathfrak{A}}_{\rho}$,
- $[\![\dot{\forall} x\, \varphi_1]\!]^{\mathfrak{A}}_{\rho} := (\forall a \in |\mathfrak{A}|)\, [\![\varphi_1]\!]^{\mathfrak{A}}_{[a/x]\rho}$ where $[a/x]\rho := i \mapsto \begin{cases} a, & \text{if } x = v_i; \\ \rho(i), & \text{otherwise.} \end{cases}$

Formalisation (Semantics)

Notations. Let $\mathfrak{A}$ be an L-structure and let $\rho : \mathbb{N} \to |\mathfrak{A}|$.

- We say that an L-formula φ ***is satisfied in*** $\mathfrak{A}$ ***under*** ρ when $[\![\varphi]\!]^{\mathfrak{A}}_{\rho}$ holds. Then write $\mathfrak{A} \models \varphi[\rho]$.
- We say that a set Γ of L-formulae ***is satisfied in*** $\mathfrak{A}$ ***under*** ρ when all of the elements of Γ are satisfied in $\mathfrak{A}$ under ρ. Then write $\mathfrak{A} \models \Gamma[\rho]$.

Definition. An L-formula C is called a ***semantic consequence*** of a set Γ of L-formulae when, for any L-structure $\mathfrak{A}$ and any $\rho : \mathbb{N} \to |\mathfrak{A}|$,

$$\text{if } \mathfrak{A} \models \Gamma[\rho] \text{ then } \mathfrak{A} \models C[\rho].$$

Then write $\Gamma \vDash C$. Otherwise, write $\Gamma \nvDash C$.

Details. For a type A, the type of subsets of A is defined as $A \to \mathbf{Prop}$—i.e.,

```
#[universes(polymorphic=yes)]
Definition ensemble@{u} (A : Type@{u}) : Type@{u} := A → Prop.
```

Thus, $\Gamma : \texttt{frm}\, L \to \mathbf{Prop}$ indicates that Γ is a set of L-formulae.

Formalisation (Deduction System)

Definition. For $\Gamma : \mathtt{frm}\, L \to \mathbf{Prop}$ and $C : \mathtt{frm}\, L$, let the proposition $\Gamma \vdash C$ mean that there exists $\vec{\varphi} : \mathtt{list}\,(\mathtt{frm}\, L)$ such that $\vec{\varphi} \subseteq \Gamma$ with $\mathrm{PF} : \mathtt{proof}\ \vec{\varphi}\ C$, where $\mathtt{list}\,(\mathtt{frm}\, L)$ is the type of finite sequences of L-formulae. Note that its negation, denoted by $\Gamma \nvdash C$, in general neither implies nor is implied by $\Gamma \vdash \dot{\neg} C$. We say that C is a ***syntactic consequence*** of Γ when $\Gamma \vdash C$ holds.

$$\textbf{Inductive}\ \mathtt{proof} : \mathtt{list}\,(\mathtt{frm}\, L) \to \mathtt{frm}\, L \to \mathbf{Set} := \cdots .$$

- Axm
$$\frac{}{\mathtt{proof}\ [p]\ p}$$
- Mp
$$\frac{\mathtt{proof}\ \vec{\varphi}_1\ (p \mathbin{\dot{\to}} q) \quad \mathtt{proof}\ \vec{\varphi}_2\ p}{\mathtt{proof}\ (\vec{\varphi}_1 +\!\!+ \vec{\varphi}_2)\ q}$$
- Gen
$$\frac{\mathtt{proof}\ \vec{\varphi}\ q}{\mathtt{proof}\ \vec{\varphi}\ (\dot{\forall} x\, q)}$$

  provided by $x \notin \mathrm{FV}(\vec{\varphi})$.

Formalisation (Deduction System)

The axiom schema for propositional logic.

- `proof` $[]\ (p \dot{\to} (q \dot{\to} p))$
- `proof` $[]\ ((p \dot{\to} (q \dot{\to} r)) \dot{\to} ((p \dot{\to} q) \dot{\to} (p \dot{\to} r)))$
- `proof` $[]\ (((\dot{\neg} q) \dot{\to} (\dot{\neg} p)) \dot{\to} (p \dot{\to} q))$

Formalisation (Deduction System)

The axiom schema for universal quantifier.

- `proof` [] $((\dot{\forall} x\, p) \dot{\to} [t/x]p)$
- `proof` [] $(p \dot{\to} (\dot{\forall} x\, p))$ provided by $x \notin \mathrm{FV}(p)$
- `proof` [] $((\dot{\forall} x\, (p \dot{\to} q)) \dot{\to} ((\dot{\forall} x\, p) \dot{\to} (\dot{\forall} x\, q)))$

Formalisation (Deduction System)

The axioms for Leibniz equality.

- $\texttt{proof}\ []\ (v_0 \doteq v_0)$
- $\texttt{proof}\ []\ ((v_0 \doteq v_1) \dot{\to} (v_1 \doteq v_0))$
- $\texttt{proof}\ []\ ((v_0 \doteq v_1) \dot{\to} ((v_1 \doteq v_2) \dot{\to} (v_0 \doteq v_2)))$
- $\texttt{proof}\ []\ (\texttt{Fun\_eqAxm}\, f)$ for each $F \in \mathcal{F}$
- $\texttt{proof}\ []\ (\texttt{Rel\_eqAxm}\, R)$ for each $R \in \mathcal{R}$

where

$$\begin{aligned}
\texttt{Fun_eqAxm}\, f \ :=\ & ((v_{2n_f-2} \doteq v_{2n_f-1}) \dot{\to} (\cdots \dot{\to} ((v_0 \doteq v_1) \dot{\to} \\
& (f(v_{2n_f-2}, \cdots, v_0) \doteq f(v_{2n_f-1}, \cdots, v_1))))), \\
\texttt{Rel_eqAxm}\, R \ :=\ & ((v_{2n_R-2} \doteq v_{2n_R-1}) \dot{\to} (\cdots \dot{\to} ((v_0 \doteq v_1) \dot{\to} \\
& (R(v_{2n_R-2}, \cdots, v_0) \dot{\to} R(v_{2n_R-1}, \cdots, v_1))))).
\end{aligned}$$

Formalisation (Meta-theory)

Theorem. The Deduction Theorem.
For any set Γ *of* L*-formulae and any* L*-formulae* A, B,

$$\Gamma \vdash A \dot{\to} B \leftrightarrow \{A\} \cup \Gamma \vdash B.$$

Proof.

($\Rightarrow$) Apply Mp and Axm.

($\Leftarrow$) There exists a finite list $\vec{\varphi}$ of L-formulae with PF : `proof` $\vec{\varphi}$ B such that $\vec{\varphi} \subseteq \{A\} \cup \Gamma$. It is sufficient to show $\vec{\varphi} \cap \Gamma \vdash A \dot{\to} B$. It can be proved by induction on PF. The most difficult case is Gen but, showing

$$A \in \vec{\varphi} \vee \vec{\varphi} \subseteq \Gamma$$

by induction on $\vec{\varphi}$, we can close the case.

□

Formalisation (Meta-theory)

Theorem. The Soundness Theorem.
For any set Γ *of* L*-formulae and any* L*-formula* C*,*

$$\Gamma \vdash C \to \Gamma \models C.$$

Proof.

$\texttt{proof}\ \vec{\varphi}\ C$ is inhabited for some $\vec{\varphi} \subseteq \Gamma$. Now, by induction on $\texttt{proof}\ \vec{\varphi}\ C$. The law of excluded middle is assumed to show the theorem, because of the following axiom scheme:

$$\texttt{proof}\ []\ (((\dot{\neg}\, q) \mathrel{\dot{\to}} (\dot{\neg}\, p)) \mathrel{\dot{\to}} (p \mathrel{\dot{\to}} q)).$$

□

Formalisation (Meta-theory)

Lemma. *For any set* Γ *of* L*-formulae and any* L*-formulae* φ_1, φ_2,

$$\frac{\Gamma \vdash \varphi_1 \quad \varphi_1 \equiv_\alpha \varphi_2}{\Gamma \vdash \varphi_2}$$

Proof.

It is enough to show

$$\varphi_1 \equiv_\alpha \varphi_2 \to (\{\varphi_1\} \vdash \varphi_2 \wedge \{\varphi_2\} \vdash \varphi_1).$$

Now, by strong induction on the height of φ_1 and `destructing` $\varphi_1 \equiv_\alpha \varphi_2$. □

Formalisation (Meta-theory)

Fact. *All of the following rules are admissible:*

$$\frac{}{\Gamma \vdash t \doteq t} \qquad \frac{\Gamma \vdash t \doteq t'}{\Gamma \vdash t' \doteq t} \qquad \frac{\Gamma \vdash t \doteq t' \quad \Gamma \vdash t' \doteq t''}{\Gamma \vdash t \doteq t''}$$

$$\frac{\Gamma \vdash t_1 \doteq t_1' \quad \cdots \quad \Gamma \vdash t_{n_f} \doteq t_{n_f}'}{\Gamma \vdash f(t_1, \cdots, t_{n_f}) \doteq f(t_1', \cdots, t_{n_f}')} \qquad \frac{\Gamma \vdash t_1 \doteq t_1' \quad \cdots \quad \Gamma \vdash t_{n_R} \doteq t_{n_R}'}{\Gamma \vdash R(t_1, \cdots, t_{n_R}) \dot{\to} R(t_1', \cdots, t_{n_R}')}$$

$$\frac{\Gamma \vdash t_1 \doteq t_2 \quad \Gamma \vdash [t_1/x]\varphi}{\Gamma \vdash [t_2/x]\varphi}$$

Proof.

Note that $\emptyset \vdash (\dot{\forall} v_{n-1}\, (\dot{\forall} v_{n-2} \cdots (\dot{\forall} v_0\, \psi))) \dot{\to} [i \mapsto \textbf{if } i < n \textbf{ then } \sigma(i) \textbf{ else } v_i]\psi$ holds for any $n \in \mathbb{N}$, any $\sigma : \mathbb{N} \to \mathtt{trm}\, L$, and any L-formula ψ. The last rule follows from

$$\frac{\Gamma \vdash t_1 \doteq t_2}{(\Gamma \vdash (\varphi[x := t_1]) \dot{\to} (\varphi[x := t_2])) \wedge (\Gamma \vdash (\varphi[x := t_2]) \dot{\to} (\varphi[x := t_1]))}$$

which can be shown by strong induction on the height of φ. □

Formalisation (Meta-theory)

Lemma. The Substitution Lemma.
For any set Γ *of* L*-formulae, any* L*-formula* φ*, and any* $\sigma : \mathbb{N} \to \mathtt{trm}\, L$*,*

$$\Gamma \vdash \varphi \to [\sigma]\Gamma \vdash [\sigma]\varphi.$$

Proof.

By the assumption $\Gamma \vdash \varphi$, there is a list $\vec{\psi}$ of L-formulae such that $\mathtt{proof}\ \vec{\psi}\ \varphi$ is inhabited and $\vec{\psi} \subseteq \Gamma$. Now, prove $\vec{\psi} \vdash \varphi \to [\sigma]\vec{\psi} \vdash [\sigma]\varphi$ by induction on $\vec{\psi}$. It is possible to prove the base case with the claim $\emptyset \vdash \varphi \to \emptyset \vdash [\sigma]\varphi$, which can be shown by induction on $\mathtt{proof}\ []\ \varphi$, and to prove the inductive case with the Deduction Theorem. □

Formalisation (Meta-theory)

Lemma. The Enumeration Lemma.

If $\mathcal{F}$, $\mathcal{C}$, and $\mathcal{R}$ are countable, then $\mathtt{trm}\, L$ *and* $\mathtt{frm}\, L$ *are enumerable.*

Proof.

Using the inverse function $\mathtt{cp} : \mathbb{N} \to (\mathbb{N} \times \mathbb{N})$ of the Cantor pairing function, one can construct functions that return abstract syntax trees of $\mathtt{trm}\, L$ and $\mathtt{frm}\, L$, respectively, with heights that are less than or equal to the second parameter, using the first parameter as the seed for AST generation. Now, using $\mathtt{cp}$ again, one can find enumerations of L-terms and L-formulae. □

In order to prove the Countable Completeness Theorem, we now assume that L is an arbitrary first-order language whose sets of function, constant, and relation symbols are countable.

Details. Let A be a type. For a surjection $f : \mathbb{N} \twoheadrightarrow A$, the sequence $\langle f(n) \rangle_{n \in \mathbb{N}}$ is called an ***enumeration*** of elements of A. A is said to be ***enumerable*** when there exists a surjection from $\mathbb{N}$ onto A. A is said to be ***countable*** when there exists an injective function from A to $\mathbb{N}$. Note that A is countable if and only if it is enumerable or empty, and that if A is enumerable then it is nonempty.

Formalisation (Meta-theory)

Definition. Each ***Henkin constant symbol*** $\overline{c}$ is defined to be a natural number. Furthermore, L' is defined to be the augmented language obtained by adding Henkin constant symbols to L.
Fact. $\mathtt{trm}\, L'$ *and* $\mathtt{frm}\, L'$ *are also enumerable.*

Proof.

Note that $\mathcal{C} \uplus \mathbb{N}$ is countable. Now, apply the Enumeration Lemma. □

Notation. For an L-formula φ, the embedding φ into L' is denoted by $\lceil\varphi$.
Fact. *For any* L*-formulae* φ *and* ψ,

$$\varphi \equiv_\alpha \psi \leftrightarrow \lceil\varphi \equiv_\alpha \lceil\psi.$$

Proof.

While the $(\Rightarrow)$ side can be proved by induction on $\varphi \equiv_\alpha \psi$, the $(\Leftarrow)$ side is implied by $(\forall p)(\forall q)(p \equiv_\alpha q \rightarrow (\forall\varphi)(\forall\psi)((p = \lceil\varphi \wedge q = \lceil\psi) \rightarrow \varphi \equiv_\alpha \psi))$, which can be shown by induction on $p \equiv_\alpha q$. □

Details. The set of constant symbols of L' can be thought of as $\mathcal{C} \uplus \mathbb{N}$, while the other sets of symbols are the same as those of L.

Formalisation (Meta-theory)

Lemma. *For any set* Γ *of* L*-formulae and any* L*-formula* φ*,*

$$\upharpoonright\Gamma \vdash \upharpoonright\varphi \leftrightarrow \Gamma \vdash \varphi.$$

Proof.

($\Leftarrow$) By induction on `proof`.

($\Rightarrow$) By the Deduction Theorem, it is sufficient to show $\emptyset \vdash \upharpoonright\varphi \to \emptyset \vdash \varphi$.
Let $\mathbf{shift}\,(X) := [v_i \mapsto v_{i+1}, \overline{c} \mapsto v_0]\,X$—i.e.,

$$\mathbf{shift}\,(v_i) := v_{i+1},\ \mathbf{shift}\,(\overline{c}) := v_0,\ \text{and}\ \mathbf{shift}\,(\dot{\forall} v_i\,\psi) := \dot{\forall} v_{i+1}\,(\mathbf{shift}\,(\psi)).$$

Then, the following can be shown by induction on `proof` $[]\ \psi$:

$$(\forall \psi \in \mathtt{frm}\,L')(\emptyset \vdash \psi \to \emptyset \vdash \mathbf{shift}\,(\psi)).$$

Now, noting $[i \mapsto v_{i-1}](\mathbf{shift}\,(\upharpoonright\varphi)) \equiv_\alpha \varphi$, it is possible to derive

$$\emptyset \vdash \upharpoonright\varphi \implies \emptyset \vdash \mathbf{shift}\,(\upharpoonright\varphi) \implies \emptyset \vdash [i \mapsto v_{i-1}](\mathbf{shift}\,(\upharpoonright\varphi)) \implies \emptyset \vdash \varphi.$$

□

Formalisation (Meta-theory)

We are going to define a sequence $\langle \theta_n \rangle_{n \in \mathbb{N}}$ of L'-formulae, which will be called the sequence of ***Henkin axioms***.

Definition. Let $\langle (x_n, \varphi_n) \rangle_{n \in \mathbb{N}}$ be a fixed enumeration of pairs, where x_n is an individual variable and φ_n is an L'-formula. For $n \in \mathbb{N}$, define

$$\theta_n := ([\overline{c}_n / x_n]\varphi_n) \dot{\to} (\dot{\forall} x_n \, \varphi_n),$$

where $\overline{c}_n$ is the first of the Henkin constant symbols not occurring in φ_n or θ_k for any $k < n$.

Details. When defining the Henkin constant $\overline{c}_n$,

I employed a technique called ***memoisation***

in order to refer to the Henkin axioms θ_k for $k < n$. Indeed, a sequence

$$\langle ((\theta_{n-1}, \theta_{n-2}, \cdots, \theta_0), (\overline{c}_{n-1}, \overline{c}_{n-2}, \cdots, \overline{c}_0)) \rangle_{n \in \mathbb{N}}$$

of pairs consisting of a vector of Henkin axioms and a vector of Henkin constants was introduced.

Formalisation (Meta-theory)

Fact. *Define* $\dot{\bot} := \dot{\neg}(\dot{\forall} v_0\,(v_0 \doteq v_0))$. *Then, for any set* Γ *of L-formulae,*

$$\Gamma \vdash \dot{\bot} \leftrightarrow \{\theta_n \mid n \in \mathbb{N}\} \cup \urcorner\Gamma \vdash \dot{\bot}\,.$$

Proof.

($\Rightarrow$) $\Gamma \vdash \dot{\bot} \implies \urcorner\Gamma \vdash \dot{\bot} \implies \{\theta_n \mid n \in \mathbb{N}\} \cup \urcorner\Gamma \vdash \dot{\bot}$.

($\Leftarrow$) Let $\Gamma_n := \{\theta_k \mid k < n\} \cup \urcorner\Gamma$. Then $\{\theta_n \mid n \in \mathbb{N}\} \cup \urcorner\Gamma \vdash \dot{\bot} \leftrightarrow (\exists n \in \mathbb{N})(\Gamma_n \vdash \dot{\bot})$. Thus, it is enough to show $(\forall n \in \mathbb{N})(\Gamma_n \vdash \dot{\bot} \to \urcorner\Gamma \vdash \dot{\bot})$. This follows from

$$\begin{aligned}
\Gamma_{n+1} \vdash \dot{\bot} &\implies \Gamma_n \vdash \dot{\neg}\,\theta_n \implies (\Gamma_n \vdash [\bar{c}_n/x_n]\varphi_n) \wedge (\Gamma_n \vdash \dot{\neg}(\dot{\forall} x_n\,\varphi_n)) \\
&\implies (\vec{\psi} \vdash [\bar{c}_n/x_n]\varphi_n) \wedge (\Gamma_n \vdash \dot{\neg}(\dot{\forall} x\,\varphi_n)) \\
&\implies ([x'/\bar{c}_n]\vec{\psi} \vdash [x'/\bar{c}_n][\bar{c}_n/x_n]\varphi_n) \wedge (\Gamma_n \vdash \dot{\neg}(\dot{\forall} x_n\,\varphi_n)) \\
&\implies (\vec{\psi} \vdash \dot{\forall} x_n\,\varphi_n) \wedge (\Gamma_n \vdash \dot{\neg}(\dot{\forall} x_n\,\varphi_n)) \implies \Gamma_n \vdash \dot{\bot},
\end{aligned}$$

where $\vec{\psi} \subseteq \Gamma_n$, $\vec{\psi} \vdash [\bar{c}_n/x_n]\varphi_n$, and $x' \notin \mathrm{FV}(\vec{\psi} +\!\!+ [\dot{\forall} x_n\,\varphi_n, [\bar{c}_n/x_n]\varphi_n])$.

□

Formalisation (Meta-theory)

Definitions. Let $\mathcal{L}$ be a first-order language.

- For a set Γ of $\mathcal{L}$-formulae, the set $\mathsf{Th}(\Gamma)$ of $\mathcal{L}$-formulae is given by

$$\mathsf{Th}(\Gamma) := \{\varphi \mid \Gamma \vdash \varphi\}.$$

  Note that Th is a closure operator on $\mathtt{frm}\,\mathcal{L}$.
- A set Γ of $\mathcal{L}$-formulae is said to be ***consistent*** when $\Gamma \nvdash \dot{\bot}$ holds.
- A set Δ of $\mathcal{L}$-formulae is said to be ***maximally consistent*** when, for any set Δ' of $\mathcal{L}$-formulae with $\Delta \subseteq \Delta'$, $\Delta' \nvdash \dot{\bot}$ if and only if $\Delta = \Delta'$.

Assuming that a consistent set Γ of L-formulae is given, let us construct a maximally consistent set Δ of L'-formulae such that

$$\{\theta_n \mid n \in \mathbb{N}\} \cup \neg\Gamma \subseteq \Delta.$$

Formalisation (Meta-theory)

Let $\langle \psi_n \rangle_{n \in \mathbb{N}}$ be a fixed enumeration of L'-formulae. Define $\langle \Delta_n \rangle_{n \in \mathbb{N}}$ by

- $\Delta_0 := \mathsf{Th}(\{\theta_n \mid n \in \mathbb{N}\} \cup \urcorner\Gamma)$,
- $\Delta_{n+1} := \mathsf{Th}(X \cup \Delta_n)$ for $n \in \mathbb{N}$ and

$$X := \left\{ b \mid b = \psi_n \wedge (\Delta_n \vdash \dot{\bot} \leftrightarrow \{\psi_n\} \cup \Delta_n \vdash \dot{\bot}) \right\}.$$

Now, take $\Delta := \bigcup_{n \in \mathbb{N}} \Delta_n$. Then Δ is maximally consistent. Hence,

$$\varphi \in \Delta \leftrightarrow \Delta \vdash \varphi$$

for any L'-formula φ. Furthermore, even without the law of excluded middle,

- for any L'-formula p, $(\dot{\neg} p) \notin \Delta \to p \in \Delta$;
- for any L'-formula p, $(\dot{\neg} p) \in \Delta \leftrightarrow p \notin \Delta$;
- for any L'-formulae p and q, $(p \mathrel{\dot{\to}} q) \in \Delta \leftrightarrow (p \in \Delta \to q \in \Delta)$; and
- for any L'-formula p and any individual variable x,

$$(\dot{\forall} x\, p) \in \Delta \leftrightarrow (\forall t \in \mathtt{trm}\, L')([t/x]p \in \Delta).$$

Formalisation (Meta-theory)

Theorem. The Model Existence Theorem.
Define an L'-structure $\mathfrak{A}$ by

- $|\mathfrak{A}| := \mathtt{trm}\, L'$,
- $t_1 \sim_{|\mathfrak{A}|} t_2 := \Delta \vdash t_1 \doteq t_2$,
- $f^{\mathfrak{A}} := \vec{t} \mapsto f\ \vec{t}$,
- $c^{\mathfrak{A}} := c$,
- $R^{\mathfrak{A}} := \vec{t} \mapsto \Delta \vdash R\ \vec{t}$,

and $\rho := i \mapsto v_i$. Then, for any L'-formula φ,

$$\varphi \in \Delta \leftrightarrow [\![\varphi]\!]^{\mathfrak{A}}_{\rho}.$$

Therefore, $\mathfrak{A} \models \Delta[\rho]$ and $\mathfrak{A} \models (\upharpoonright\Gamma)[\rho]$.

Proof.

By strong induction on the height of φ. This theorem can be proved without the law of excluded middle. □

Formalisation (Meta-theory)

Theorem. The Completeness Theorem (for Every Countable Language).
Let X be an arbitrary set of L-formulae and let an L-formula b be an arbitrary semantic consequence of X. Then $X \vdash b$.

Proof.

Assume $X \nvdash b$. Now, set $\Gamma := \{\dot{\neg} b\} \cup X$. Then, we have $\Gamma \nvdash \dot{\bot}$. Restricting the L'-structure obtained by the Model Existence Theorem to L yields $\Gamma \nvDash \dot{\bot}$, which contradicts the assumption $X \vDash b$. Therefore, we can conclude that the assumption $X \nvdash b$ is false and finally obtain $X \vdash b$. □

Comparison

Comparison with Ilik (2010).
Dr. Danko Ilik formalised in the first chapter of his thesis the countable completeness Theorem of classical natural deduction, but the proof was incomplete. He acknowledged that there was a lemma admitted without formal proof.

Comparison

Comparison with Herberlin, Kim, and Lee (2017).
Dr. Hugo Herberlin, Prof. SunYoung Kim, and Prof. Gyesik Lee formalised the weak completeness theorem of the Gentzen-style sequent calculus LJT for Kripke's semantics instead of Tarski's definition of truth.

Comparison

Comparison with Forster, Kirst, and Wehr (2021).
Prof. Yannick Forster, Dr. Dominik Kirst, and Dominik Wehr formalised the completeness theorem of classical natural deduction. However, their setting, based on de Bruijn index, makes it difficult to use as a framework.

Comparison

Comparison with From (2022).
Dr. Asta Halkjær From formalised the completeness theorem of a Hilbert calculus while using de Bruijn index. However, there is a side condition in the statement of the completeness theorem. The main theorem of the study is

$$\emptyset \vdash \varphi \leftrightarrow \emptyset \vDash \varphi.$$

Comparison

Comparison with Herberlin and Ilik (2024).
Dr. Hugo Herbelin and Dr. Danko Ilik formalised the completeness theorem for classical first-order languages *not* equipped with Leibniz equality. They did not introduce Henkin constant symbols and modified Henkin's method.

Conclusion

The Main Result. I formalised classical first-order logic *equipped with* Leibniz equality using `Coq 8.18.0`, *assuming only the law of excluded middle.*

```
1 Check @HilbertCalculus_sound.
2 Print Assumptions HilbertCalculus_sound.
3 Check @HilbertCalculus_complete.
4 Print Assumptions HilbertCalculus_complete.
```

Figure 1: A Coq Script for Checking Theorem Statements and Used Axioms.

Conclusion

```
@HilbertCalculus_sound
     : forall (L : language) (Gamma : ensemble (frm L)) (C : frm L),
       Gamma ⊢ C -> Gamma ⊨ C
Axioms:
classic : forall P : Prop, P \/ ~ P
@HilbertCalculus_complete
     : forall L : language,
       isCountable (function_symbols L) ->
       isCountable (constant_symbols L) ->
       isCountable (relation_symbols L) ->
       forall (X : ensemble (frm L)) (b : frm L), X ⊨ b -> X ⊢ b
Axioms:
classic : forall P : Prop, P \/ ~ P
lim@K1-20230101CTCH:~/portfolio/Fol-archived$
```

Figure 2: The Result from the Coq Script.

Conclusion

The Contributions.

- The lemma that, for any set Γ of L-formulae and any L-formula φ,

$$\gimel\Gamma \vdash \gimel\varphi \leftrightarrow \Gamma \vdash \varphi,$$

  had not been formalised in any former studies.
- My formal proof of the Countable Completeness Theorem closely follows the approach presented in Enderton's mathematical text, ensuring its fidelity to orthodox methodology.
- While there are many studies that formalise natural deduction, Hilbert calculi have received comparatively less attention. This makes my formalisation a significant contribution.

Conclusion

Future Work.

- The practical applicability of the framework such as PA and ZFC will be explored.
- Since both systems have axiom schemata, a tool will be made for handling meta-variables.
- The Completeness Theorem for first-order languages with cardinalities greater than $\aleph_0$ will be formally proved as well.

References

[1] Robert Constable and Mark Bickford. "Intuitionistic Completeness of First-Order Logic". In: *Annals of Pure and Applied Logic* 165.1 (2014). The Constructive in Logic and Applications, pp. 164–198. ISSN: 0168-0072. DOI: 10.1016/j.apal.2013.07.009. URL: https://www.sciencedirect.com/science/article/pii/S0168007213001085.

[2] Ernesto Copello, Nora Szasz, and Álvaro Tasistro. "Formal Metatheory of the Lambda Calculus Using Stoughton's Substitution". In: *Theoretical Computer Science* 685 (2017). Logical and Semantic Frameworks with Applications, pp. 65–82. ISSN: 0304-3975. DOI: 10.1016/j.tcs.2016.08.025. URL: https://www.sciencedirect.com/science/article/pii/S0304397516304820.

[3] Herbert B. Enderton. *A Mathematical Introduction to Logic*. Academic Press, 2001. ISBN: 978-0-12-238452-3.

[4] Yannick Forster, Dominik Kirst, and Dominik Wehr. "Completeness theorems for first-order logic analysed in constructive type theory: Extended version". In: *Journal of Logic and Computation* 31.1 (Jan. 2021), pp. 112–151. ISSN: 0955-792X. DOI: 10.1093/logcom/exaa073. eprint: https://academic.oup.com/logcom/article-pdf/31/1/112/36719784/exaa073.pdf.

[5] Asta Halkjær From. "A Succinct Formalization of the Completeness of First-Order Logic". In: *Proceedings of the 27th International Conference on Types for Proofs and Programs (TYPES 2021)*. Vol. 239. Leibniz International Proceedings in Informatics (LIPIcs). Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022, 8:1–8:24. DOI: 10.4230/LIPIcs.TYPES.2021.8.

[6] Hugo Herbelin and Danko Ilik. *An analysis of the constructive content of Henkin's proof of Gödel's completeness theorem*. arXiv:2401.13304. 2024. arXiv: 2401.13304 [math.LO]. URL: https://arxiv.org/abs/2401.13304.

[7] Hugo Herberlin, SunYoung Kim, and Gyesik Lee. "FORMALIZING THE META-THEORY OF FIRST-ORDER PREDICATE LOGIC". In: *Journal of the Korean Mathematical Society* 54.5 (Sept. 2017). 대한수학회, pp. 1521–1536. DOI: 10.4134/JKMS.J160546.

[8] Danko Ilik. "Constructive Completeness Proofs and Delimited Control. (Preuves constructives de complétude et contrôle délimité)". PhD thesis. École Polytechnique, Palaiseau, France, 2010. URL: https://tel.archives-ouvertes.fr/tel-00529021.

[9] Ralf Jung et al. "Iris: Monoids and Invariants as an Orthogonal Basis for Concurrent Reasoning". In: *SIGPLAN Not.* 50.1 (Jan. 2015), pp. 637–650. ISSN: 0362-1340. DOI: 10.1145/2775051.2676980. URL: https://doi.org/10.1145/2775051.2676980.

[10] M. Kaufmann and J. Strother Moore. "ACL2: an industrial strength version of Nqthm". In: *Proceedings of 11th Annual Conference on Computer Assurance. COMPASS '96*. 1996, pp. 23–34. DOI: 10.1109/CMPASS.1996.507872.

[11] Peter Lammich and Simon Wimmer. "IMP2 – Simple Program Verification in Isabelle/HOL". In: *Archive of Formal Proofs* (Jan. 2019). https://isa-afp.org/entries/IMP2.html, Formal proof development. ISSN: 2150-914x.

[12] Xavier Leroy et al. "CompCert – A Formally Verified Optimizing Compiler". In: *ERTS 2016: Embedded Real Time Software and Systems*. SEE, 2016. URL: http://xavierleroy.org/publi/erts2016_compcert.pdf.

[13] Kijeong Lim. *Fol-archived*. Available at: https://github.com/KiJeong-Lim/Fol-archived.

[14] Russell O'Connor. "Incompleteness & completeness: formalizing logic and analysis in type theory". PhD thesis. Netherlands: Radboud University Nijmegen, 2009.

[15] Youngju Song et al. "Conditional Contextual Refinement". In: *Proc. ACM Program. Lang.* 7.POPL (Jan. 2023). DOI: 10.1145/3571232. URL: https://doi.org/10.1145/3571232.

Thank you for listening!

E-mail: gijungdduk@naver.com

GitHub: https://github.com/PnVDiscord/PnVRocqLib

Archive: https://github.com/KiJeong-Lim/Fol-archived