

逻辑学讨论课

类型论

杨睿之

复旦大学哲学学院

2025 年秋季

罗素悖论

考虑罗素集：

$$R = \{x \mid x \notin x\}$$

问：是否 $R \in R$ ？

罗素悖论

- 集合论是一种 **untyped** 理论
- 类型论中每个对象都有类型，例如： $a : \text{Set}$ 、 $b : \text{Set}$ ，
而 $\text{Set} : \text{Type}$
- 类型论不允许任何 $x : x$ ，那类型的类型呢？
 - 类型的层级： $U_0 : U_1$ 、 $U_1 : U_2$
- 我们可以有 $R = \{x : U_0 \mid x \notin x\} : U_1$ 。注意 $\in : U_0 \times U_0$ ，
当我们问是否 $R \in R$ ，我们问的 $\in : U_1 \times U_1$ 。

构造主义

- 所有数学对象都是基于规则被构造出来的
 - 不允许非直谓的构造
 - 例：罗素集
- 真是被证明

构造主义

Brouwer–Heyting–Kolmogorov interpretation (BHK-解释)

- 一个 $P \wedge Q$ 的证明 是一个有序对 (a, b) , 其中, a 是 P 的一个证明, b 是 Q 的一个证明
- 一个 $P \vee Q$ 的证明 是一个 P 的证明或一个 Q 的证明
- 一个 $P \rightarrow Q$ 的证明 是一个构造, 它可以把任何一个 P 的证明转化为一个 Q 的证明

构造主义

Brouwer–Heyting–Kolmogorov interpretation (BHK-解释)

- 一个 $(\exists x \in S)(Px)$ 的证明 是一个有序对 (x, a) , 其中 x 是 S 中的元素, 且 a 是一个 Px 的证明
- 一个 $(\forall x \in S)(Px)$ 的证明 是一个构造, 它可以把任何一个 S 中的 x 转化为一个 Px 的证明
- 一个 $\neg P$ (被定义为 $P \rightarrow \perp$) 的证明 是一个构造, 它可以把任何一个 P 的证明转化为有一个对 \perp 的证明
- 不存在 \perp (荒谬) 的证明

构造主义

- **排中律** ($P \vee \neg P$) 不总是成立
- **双重否定消去** ($\neg\neg P \rightarrow P$) 不总是成立, 反证法不能任意使用。
但 $P \rightarrow \neg\neg P$ 成立, 你能证明吗?
- 经典 (集合论的) **选择公理** 不成立

寻求可实践的数学基础

- 数学证明的验证危机
- 证明辅助
- 沟通人与机器的语言
数学工业化？

数学证明的验证危机

Kepler 猜想

- Thomas Hales 在 1998 年宣布了一个证明，该证明含有 250 页笔记和 3GB 的计算机程序、数据和结论
- *Annals of Mathematics* 组织了审稿，最终在 2003 发表，审稿意见是：“99% certain”
- Thomas Hales 团队在 2014 年宣布了一个形式化证明（使用 HOL Light 和 Isabelle）
- 2017 年，该形式证明被接受发表于 *Forum of Mathematics*

数学证明的验证危机

但更多的人没有 Thomas Hales 那么幸运

望月新一关于 ABC 猜想的工作

- 望月新一在 2012 年宣布使用“Inter-universal Teichmüller”理论证明了 ABC 猜想，证明有 750 余页
- 2018 年 Scholze 和 Stix 访问望月新一，经过长时间的讨论，撰写了报告 *Why abc is still a conjecture*
- 2021 年望月新一的证明在 RIMS 发表，但.....

证明辅助

Peter Scholze's Liquid Tensor Experiment

- 2020 年, Peter Scholze 提出了一项挑战, 要求形式化验证他和 Dustin Clausen 的 theory of condensed mathematic 中的一个核心且复杂的定理。由于该证明极其复杂, Scholze 并不完全确定其正确性。

证明辅助

I spent much of 2019 obsessed with the proof of this theorem, almost getting crazy over it. In the end, we were able to get an argument pinned down on paper, but I think nobody else has dared to look at the details of this, and so I still have some small lingering doubts.

(Scholze, [Liquid tensor experiment](#))

证明辅助

with this theorem, the hope that the condensed formalism can be fruitfully applied to real functional analysis stands or falls. I think the theorem is of utmost foundational importance, so being 99.9% sure is not enough.

(Scholze, [Liquid tensor experiment](#))

证明辅助

Peter Scholze's Liquid Tensor Experiment

- 半年后的 2021 年 5 月，定理 9.4 的形式化证明公布，标志着该项目第一部分的完成。
- 一年半后，Lean 社区宣布该项目完成。其中还包含一份便于人类阅读的[蓝图](#)。

证明辅助

Theorem 9.4 is an extremely technical statement, whose proof is however the heart of the challenge, and is the only result I was worried about. So with its formal verification, I have no remaining doubts about the correctness of the main proof.

(Scholze, Half a year)

证明辅助

The Lean Proof Assistant was really that: An assistant in navigating through the thick jungle that this proof is. Really, one key problem I had when I was trying to find this proof was that I was essentially unable to keep all the objects in my “RAM”, and I think the same problem occurs when trying to read the proof.

(Scholze, Half a year)

证明辅助

Lean always gives you a clear formulation of the current goal, and Johan confirmed to me that when he formalized the proof of Theorem 9.4, he could —with the help of Lean —really only see one or two steps ahead, formalize those, and then proceed to the next step. So I think here we have witnessed an experiment where the proof assistant has actually assisted in understanding the proof.

(Scholze, *Half a year*)

证明辅助

Actually, there would be a very clear thing that might have gone wrong in this: Namely, that the formalized theorem statement isn' t quite what you think it is. Checking that it is indeed the correct statement is quite tedious —as I said Theorem 9.4 is really quite a technical statement, and to check that all definitions that enter it are exactly correct is still something that a human must do, and still a tedious task.

(Scholze, *Half a year*)

证明辅助

The Lean Proof Assistant was really that: An assistant in navigating through the thick jungle that this proof is. Really, one key problem I had when I was trying to find this proof was that I was essentially unable to keep all the objects in my "RAM" , and I think the same problem occurs when trying to read the proof.

(Scholze, *Half a year*)

证明辅助

Lean always gives you a clear formulation of the current goal, and Johan confirmed to me that when he formalized the proof of Theorem 9.4, he could —with the help of Lean —really only see one or two steps ahead, formalize those, and then proceed to the next step. So I think here we have witnessed an experiment where the proof assistant has actually assisted in understanding the proof.

(Scholze, *Half a year*)

证明辅助

Fermat's Last Theorem (FLT, 费马大定理)

- 第一个被广泛接受的证明是由 Andrew Wiles 爵士于 1993 年宣布的。
- 原始证明使用了 Grothendieck's universe, 这相当于存在一个不可达基数
- 该证明使用了大量 20 世纪精妙的数学技巧

证明辅助

关于 FLT 的项目：

- 在一个弱的算术系统中证明 FLT (反推数学)
- 把 FLT 定理的证明归约于那些 80 年代前人们所熟知的数学 ([The FLT Project in Lean community](#))

沟通人与机器的语言

Formal proof assistants can be used to verify proofs (as well as the output of large language models), allow truly large-scale mathematical collaborations, and help build data sets to train the aforementioned machine learning algorithms.

(Tao, AMS Colloquium Lectures)

沟通人与机器的语言

One notable feature of proof formalization projects is that they lend themselves to large collaborations that do not require high pre-established levels of trust.

(Tao, AMS Colloquium Lectures)

类型论

类型论是下面这些证明辅助语言的基础

- RCoq
- Lean
- Agda

类型论

类型论的核心想法

- 命题即类型
- 证明即由构造例证
- 构造基于规则

类型论

类型论的两种基本判断 (judgement)

$a : A$ (类型正确 (well-typed))

$a \equiv b : A$ (定义等同 (definitional equal))

类型论

- 定义等同

Example: $f(x) \equiv x + x : \mathbb{N}$ (\equiv 是元语言符号)

- 命题等同 (Propositional equality)

Example: $p : a =_A b$