Mechanised Metamathematics

An Investigation of First-Order Logic and Set Theory in Constructive Type Theory

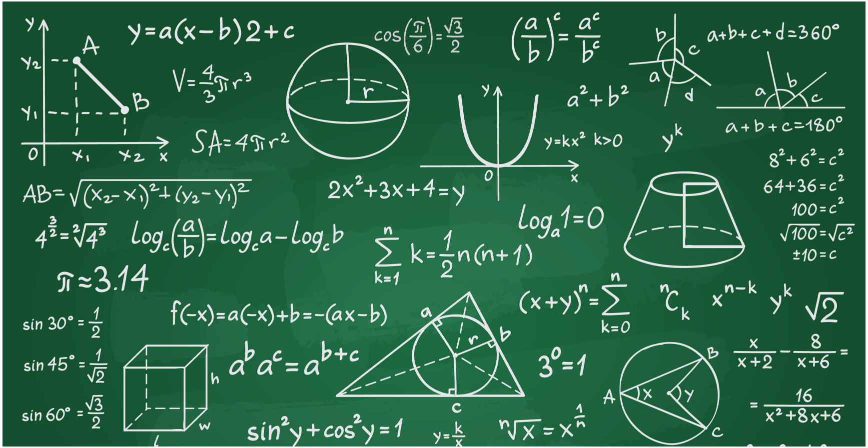
Dominik Kirst

Thesis Colloquium Talk, January 27th, 2023

SAARLAND UNIVERSITY
COMPUTER SCIENCE

Slogans and Images

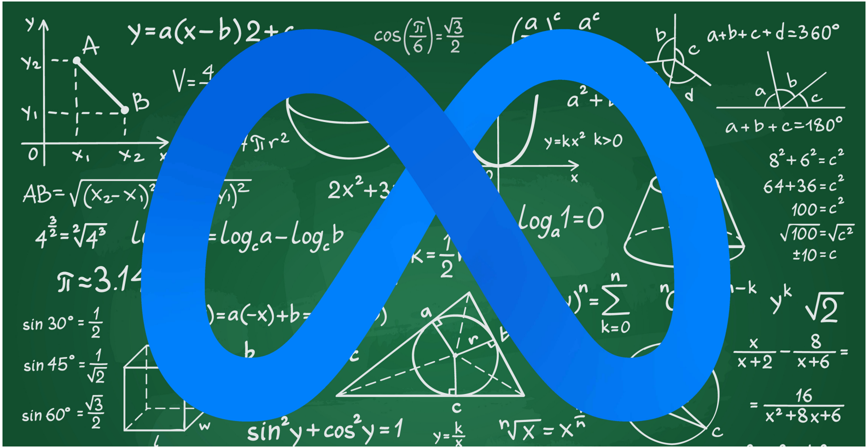
What is Mathematics?



https://msutexas.edu/academics/scienceandmath/mathematics/_assets/images/mathematics_image.jpg

Example: "There are infinitely many prime numbers"

What is Metamathematics?



https://1000logos.net/wp-content/uploads/2021/10/logo-Meta.png

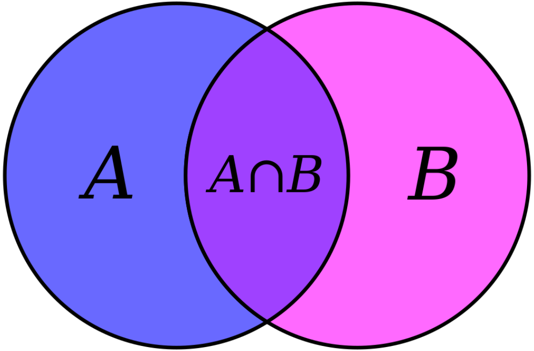
“The proposition ’there are infinitely many prime numbers’ is provable in number theory”

What is First-Order Logic?

https://upload.wikimedia.org/wikipedia/commons/thumb/4/4a/Emblem_of_the_First_Order.svg/888px-Emblem_of_the_First_Order.svg.png

Universal language consisting of a formal syntax, semantics, and deduction system

What is Set Theory?



https://upload.wikimedia.org/wikipedia/commons/thumb/6/6d/Venn_A_intersect_B.svg/1200px-Venn_A_intersect_B.svg.png

Foundational first-order axiom system that can express all of mathematics

What is Type Theory?

https://http2.mlstatic.com/D_NQ_NP_2X_731038-MLM47969124431_102021-F.jpg

An alternative foundation emphasising structure and interrelations of objects

What is Constructive Logic?

https://img.redbull.com/images/c_limit,w_1500,h_1000,f_auto,q_auto/redbullcom/2020/4/28/bjoyslzjb3uxqyg82uz2/minecraft

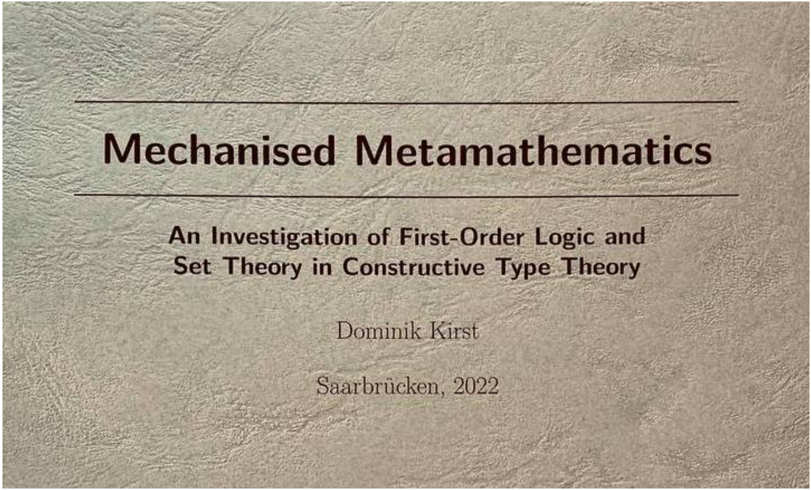
An alternative logic emphasising construction and computation

What is Computer Mechanisation?

https://ilyasergey.net/pnp/coq-logo.png

Usage of interactive computer systems to formulate, verify, and automate mathematical proofs

So what is the thesis about?



The Canon of First-Order Logic

Completeness, Undecidability, Incompleteness

The Calculus of Inductive Constructions (CIC)

Main features of CIC (Coquand and Huet, 1988; Paulin-Mohring, 1993):

- Typing judgement $x : X$ prescribes type X to term x, e.g. $5 : \mathbb{N}$ or $(\lambda n.\, n + 1) : \mathbb{N} \to \mathbb{N}$
- Simple and dependent type formers: $X \to Y$, $X \times Y$, $X + Y$, $\forall x.\, F\, x$, $\Sigma x.\, F\, x$
- Inductive types: $\mathbb{B}$, $\mathbb{N}$, lists $\mathbb{L}(X)$, options $\mathbb{O}(X)$, vectors X^n, ...
- Propositional universe $\mathbb{P}$ with logical connectives: $\to$, $\wedge$, $\vee$, $\forall$, $\exists$
- $\mathbb{P}$ is impredicative and (almost) disconnected from computational types

The internal logic of $\mathbb{P}$ is intuitionistic, e.g. the following classical principles are unprovable:

- Law of excluded middle (LEM): $\forall P : \mathbb{P}.\, P \vee \neg P$
- Markov's principle (MP): $\forall f : \mathbb{N} \to \mathbb{B}.\, \neg\neg(\exists n.\, f\, n = \mathsf{tt}) \to \exists n.\, f\, n = \mathsf{tt}$
- There are non-computable functions

Representing First-Order Logic in Constructive Type Theory

Terms and formulas are represented as inductive types $\mathbb{T}$ and $\mathbb{F}$ over a signature $\Sigma = (\mathcal{F}_\Sigma, \mathcal{P}_\Sigma)$:

$$t : \mathbb{T} \;::=\; x \mid f\,\vec{t} \qquad (x : \mathbb{N}, f : \mathcal{F}_\Sigma, \vec{t} : \mathbb{T}^{|f|})$$

$$\varphi, \psi : \mathbb{F} \;::=\; \bot \mid P\,\vec{t} \mid \varphi \to \psi \mid \varphi \land \psi \mid \varphi \lor \psi \mid \forall x.\,\varphi \mid \exists x.\,\varphi \qquad (P : \mathcal{P}_\Sigma, \vec{t} : \mathbb{T}^{|P|})$$

- Natural deduction ($\Gamma \vdash \varphi$) captured by inductive rules of intuitionistic or classical flavour
- Tarski semanticts ($\Gamma \vDash \varphi$) defined recursively over models providing enough structure
- Axioms systems like PA and ZF induce relativised theories, e.g. PA $\vdash \varphi$ and PA $\vDash \varphi$

Constructive Completeness[1]

In which situations does $\Gamma \vDash \varphi$ imply $\Gamma \vdash \varphi$?

- Gödel: completeness holds (Gödel, 1930)
- Also Gödel: completeness does not hold constructively (Kreisel, 1962)
- Constructive completeness desirable: executable reification of meta-level proof terms
- Rich (and confusing) literature on constructive reverse mathematics of completeness
- We were mostly inspired by Herbelin and Ilik (2016) and Herbelin and Lee (2009)

[1] Forster, Kirst, and Wehr (2021)

Model-Theoretic Semantics

Theorem (Quasi-Completeness)

In the negative $(\to, \forall, \bot)$-fragment, assuming $\Gamma \vDash \varphi$ implies that $\Gamma \vdash \varphi$ does not not hold.

Theorem

In the minimal $(\to, \forall)$-fragment, $\Gamma \vDash \varphi$ implies $\Gamma \vdash \varphi$. However, including $\bot$ one observes:

1. *Completeness for enumerable contexts Γ is equivalent to* MP,
2. *Completeness for arbitrary contexts Γ is equivalent to* LEM.

- Constructive for relaxed interpretation of $\bot$ (Veldman, 1976)
- Similar results relating Kripke semantics $\Gamma \Vdash \varphi$ with intuitionistic deduction $\Gamma \vdash_i \varphi$
- Fully constructive completeness for algebraic semantics (Scott, 2008)

The Case of Disjunctions (and Existentials)

Constructivising the metatheory of intuituitionistic epistemic logic (Hagemeier and Kirst, 2022), a propositional modal logic including $\vee$, we observed the following connections:

Fact (Intuitionistic Epistemic Logic)

1. *Model Existence is equivalent to WLEM:* $\forall P : \mathbb{P}.\, \neg P \vee \neg\neg P$
2. *Quasi-Completeness is derivable from DNS:* $\forall X. \forall p : X \to \mathbb{P}.\, (\forall x.\, \neg\neg p\, x) \to \neg\neg(\forall x.\, p\, x)$
3. *Quasi-Completeness implies a principle we call WDNS:* $\forall p : \mathbb{N} \to \mathbb{P}.\, \neg\neg(\forall n.\, \neg p\, n \vee \neg\neg p\, n)$
4. *Not included in the thesis: Quasi-Completeness is derivable from WDNS*

We expect the new observations to apply to first-order logic including existentials, therefore:

Conjecture (First-Order Logic)

Quasi-Completeness for the full syntax is equivalent to WDNS.

Synthetic Undecidability[2]

Which decision problems of first-order logic are undecidable?

- Church/Turing: validity and provability are undecidable (Church, 1936; Turing, 1937)
- Proofs by computable reduction, referring to an explicit model of computation
- Synthetic computability avoids explicit models (Richman, 1983; Bauer, 2006)
- Synthetic undecidability proofs feasible to mechanise (Forster, 2021; Forster et al., 2020)
- Outline: define and verify reduction functions in constructive logic

[2] Forster, Kirst, and Smolka (2019)

The Entscheidungsproblem

Use the Post correspondence problem (PCP) as seed, with instances $S : \mathbb{L}(\mathbb{B}^* \times \mathbb{B}^*)$:

$$\frac{(s,t) \in S}{S \rhd (s,t)} \qquad \frac{S \rhd (u,v) \quad (s,t) \in S}{S \rhd (su, tv)} \qquad \frac{S \rhd (s,s)}{\text{PCP } S}$$

Encode instances S of PCP as formulas φ_S over signature $(e, f_{\mathsf{tt}}\_, f_{\mathsf{ff}}\_; Q, P\_\_)$:

$$\begin{aligned} \overline{\epsilon} &:= e & \overline{bs} &:= f_b(\overline{s}) \\ \varphi_1 &:= [P\,\overline{s}\,\overline{t} \mid (s,t) \in S] & \varphi_2 &:= [\forall xy.\, P\,x\,y \to P\,(\overline{s}x)\,(\overline{t}y) \mid (s,t) \in S] \\ \varphi_3 &:= \forall x.\, P\,x\,x \to Q & \varphi_S &:= \varphi_1 \to \varphi_2 \to \varphi_3 \to Q \end{aligned}$$

Verify that this translation describes a reduction function:

Theorem

PCP S *iff* φ_S *is valid, thus* PCP *reduces to validity.*

Variants of the Entscheidungsproblem

Using the same synthetic method, we obtain undecidability of:

- Satisfiability in Tarski semantics
- Provability in intuitionistic and classical natural deduction
- Validity and satisfiability in Kripke semantics
- Trakhtenbrot's theorem: finite Tarski satisfiability (Kirst and Larchey-Wendling, 2022)
- All the above restricted to binary signature (Hostert, Dudenhefner, and Kirst, 2022)
- Several fragments of Robinson's Q, PA, and ZF (Kirst and Hermes, 2021)

Based on mechanised undecidability proofs of:

- Post correspondence problem (Forster et al., 2018)
- Solvability of Diophantine equations (Larchey-Wendling and Forster, 2019)

Synthetic Incompleteness[3]

Which axiom systems $\mathcal{A}$ satisfy $\mathcal{A} \vdash \varphi$ or $\mathcal{A} \vdash \neg\varphi$ for all φ?

- Gödel: all sound, sufficiently expressive ones (Gödel, 1931)
- Rosser: all consistent, sufficiently expressive ones (Rosser, 1936)
- Post/Church/Turing: Gödel's incompleteness is a consequence of undecidability
- Kleene: Rosser's incompleteness is a consequence of recursive inseparability
- We give synthetic computational proofs complementing mechanisations à la Gödel/Rosser: Shankar (1986); O'Connor (2005); Paulson (2015); Popescu and Traytel (2021)

[3] Kirst and Peters (2023)

Synthetic Church-Turing

Fact

If Robinson's Q *(or any sound extension) is complete, then the halting problem is decidable.*

Improvement assuming a form of Church's thesis (Richman, 1983; Forster, 2022):

Axiom (EPF)

There is a universal function $\Theta : \mathbb{N} \to (\mathbb{N} \rightharpoonup \mathbb{N})$ *enumerating all partial functions:*

$$\forall f : \mathbb{N} \rightharpoonup \mathbb{N}. \exists c : \mathbb{N}. \forall xy. \Theta_c\, x \downarrow y \leftrightarrow f\, x \downarrow y$$

Theorem

Every axiom system $\mathcal{A}$ *representing* $\mathsf{K}\, x := \Theta_x\, x \downarrow$*, i.e. providing* φ_K *with*

$$\mathsf{K}\, x \;\leftrightarrow\; \mathcal{A} \vdash \varphi_\mathsf{K}(\overline{x})$$

neither proves nor refutes $\varphi_\mathsf{K}(\overline{c})$ *for* c *being the code of a diagonalisation against* K.

Gödel, Rosser, Kleene

Following an idea of Kleene (1951), we derive a stronger version:

Theorem

Every axiom system $\mathcal{A}$ separating $\mathsf{K}^1\, x := \Theta_x\, x \downarrow 1$ and $\mathsf{K}^0\, x := \Theta_x\, x \downarrow 0$, i.e. providing φ_K with

$$\mathsf{K}^1\, x \to \mathcal{A} \vdash \varphi_\mathsf{K}(\overline{x}) \qquad \text{and} \qquad \mathsf{K}^0\, x \to \mathcal{A} \vdash \neg\varphi_\mathsf{K}(\overline{x})$$

neither proves nor refutes $\varphi_\mathsf{K}(\overline{c})$ for c being the code of a diagonalisation against K^1 and K^0.

To instantiate these abstract proofs to Q, we need a stronger assumption than EPF:

Axiom ($\mathsf{CT_Q}$, cf. Hermes and Kirst (2022))

For every $f : \mathbb{N} \rightharpoonup \mathbb{N}$ there exists $\varphi(x, y)$ with: $\forall xy.\, f\, x \downarrow y \;\leftrightarrow\; \mathsf{Q} \vdash \forall y'.\, \varphi(\overline{x}, y') \leftrightarrow y' = \overline{y}$

Theorem

$\mathsf{CT_Q}$ *implies* EPF *and that* Q *separates the respective problems* K^1 *and* K^0.
Thus every consistent extension of Q *admits an independent sentence.*

A Coq Library for Mechanised First-Order Logic[4]

`https://github.com/uds-psl/coq-library-fol`

- Merge of all developments into continuously developed core library
- Meant to serve as general framework for future projects, also by external users

- Only well-formed terms and formulas using vectors to implement symbol arities
- Modularity by (type class) parameters for signatures, connectives, and deduction rules
- Mechanised de Bruijn encoding inspired by Autosubst 2 (Stark et al., 2019; Stark, 2019)
- Tool support for syntax, deduction, and semantics (Hostert, Koch, and Kirst, 2021)

[4] Kirst et al. (2022)

Three Levels of Set Theory

First-Order, Second-Order, Synthetic

Three Levels of Set Theory in CIC[5]

	First-Order	Second-Order	Synthetic
Power sets	$\mathcal{P}(A)$		$X \to \mathbb{P}$
Numbers	ω	-	$\mathbb{N}$
Relations	$\mathcal{P}(A \times B)$	both coincide	$X \to Y \to \mathbb{P}$
Functions	$\{f \subseteq A \times B \mid \dots\}$	-	$X \to Y$
Cardinality	$\exists f \subseteq A \times B \dots$		$\exists f : X \to Y \dots$
Orderings	$\exists R \subseteq A \times A \dots$		$\exists R : X \to X \to \mathbb{P} \dots$

[5] Kirst and Hermes (2021); Kirst and Smolka (2019); Kirst and Rech (2021)

Case Study: Sierpiński's Theorem

The Generalised Continuum Hypothesis implies the Axiom of Choice (Sierpiński, 1947)

$$\text{GCH} \;:=\; \forall AB.\, |\mathbb{N}| \leq |A| \leq |B| \leq |\mathcal{P}(A)| \rightarrow |B| \leq |A| \,\vee\, |\mathcal{P}(A)| \leq |B|$$

$$\text{AC} \;:=\; \forall AB.\, \forall R \subseteq A \times B.\, (\forall x.\, \exists y.\, R\,x\,y) \rightarrow \exists f : A \rightarrow B.\, \forall x.\, R\,x\,(f\,x)$$

- Given A, construct a well-ordered set $\aleph(A)$ with $|\aleph(A)| \not\leq |A|$ but $|\aleph(A)| \leq |\mathcal{P}^6(A)|$
- Iterate GCH to obtain $|A| \leq |\aleph(A)|$, thus A can be well-ordered and satisfies AC
- May use LEM since already weak forms of GCH imply LEM

- FOL: hard work done by Carneiro (2015)
- SOL: simpler mechanisation, delegating cardinal arithmetic to type level
- CIC: construction of $\aleph(A)$ circumventing ordinal theory
- HoTT: natural combination of set-theoretic techniques with type-theoretic primitives

Conclusion

Contributions

- Formalisation: uniform development of metamathematics in constructive type theory
- Mechanisation: reusable Coq libraries advancing original goals of metamathematics
- Constructivisation: fully constructive where possible, sharply analysed otherwise
- Simplification: synthetic method streamlines undecidability and incompleteness proofs
- Orientation: accessible and modern overview of the standard canon of metamathematics

Perspectives

There's a lot of things I plan to continue working on:

- What is the constructive status of completeness theorems, really?
- Does the synthetic method help with Gödel's second incompleteness theorem?
- Is there a natural description of the constructible hierarchy in constructive type theory?
- ...

Thank you all, for everything!

Bibliography I

Bauer, A. (2006). First steps in synthetic computability theory. *Electronic Notes in Theoretical Computer Science*, 155:5–31.

Carneiro, M. (2015). GCH implies AC, a Metamath Formalization. In *8th Conference on Intelligent Computer Mathematics*, Workshop on Formal Mathematics for Mathematicians.

Church, A. (1936). A note on the Entscheidungsproblem. *The journal of symbolic logic*, 1(1):40–41.

Coquand, T. and Huet, G. (1988). The calculus of constructions. *Information and Computation*, 76(2):95–120.

Forster, Y. (2021). *Computability in constructive type theory*. PhD thesis, Saarland University. `https://www.ps.uni-saarland.de/~forster/thesis.php`.

Forster, Y. (2022). Parametric Church's thesis: Synthetic computability without choice. In *International Symposium on Logical Foundations of Computer Science*, pages 70–89. Springer.

Forster, Y., Heiter, E., and Smolka, G. (2018). Verification of PCP-related computational reductions in Coq. In *International Conference on Interactive Theorem Proving*, pages 253–269. Springer.

Forster, Y., Kirst, D., and Smolka, G. (2019). On synthetic undecidability in Coq, with an application to the Entscheidungsproblem. In *International Conference on Certified Programs and Proofs*. ACM.

Forster, Y., Kirst, D., and Wehr, D. (2021). Completeness theorems for first-order logic analysed in constructive type theory: Extended version. *Journal of Logic and Computation*, 31(1):112–151.

Bibliography II

Forster, Y., Larchey-Wendling, D., Dudenhefner, A., Heiter, E., Kirst, D., Kunze, F., Smolka, G., Spies, S., Wehr, D., and Wuttke, M. (2020). A Coq library of undecidable problems. In *CoqPL Workshop*.

Gödel, K. (1931). Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I. *Monatshefte für mathematik und physik*, 38(1):173–198.

Gödel, K. (1930). Die Vollständigkeit der Axiome des logischen Funktionenkalküls. *Monatshefte für Mathematik und Physik*, 37:349–360.

Hagemeier, C. and Kirst, D. (2022). Constructive and mechanised meta-theory of IEL and similar modal logics. *Journal of Logic and Computation*.

Henkin, L. (1954). Metamathematical theorems equivalent to the prime ideal theorem for boolean algebras. *Bulletin AMS*, 60:387–388.

Herbelin, H. (2022). Computing with gödel's completeness theorem: Weak fan theorem, markov's principle and double negation shift in action. `http://pauillac.inria.fr/~herbelin/talks/chocola22.pdf`.

Herbelin, H. and Ilik, D. (2016). An analysis of the constructive content of Henkin's proof of Gödel's completeness theorem. Draft.

Herbelin, H. and Lee, G. (2009). Forcing-based cut-elimination for Gentzen-style intuitionistic sequent calculus. In *International Workshop on Logic, Language, Information, and Computation*, pages 209–217. Springer.

Bibliography III

Hermes, M. and Kirst, D. (2022). An analysis of Tennenbaum's theorem in constructive type theory. In *International Conference on Formal Structures for Computation and Deduction*. LIPIcs.

Hostert, J., Dudenhefner, A., and Kirst, D. (2022). Undecidability of dyadic first-order logic in Coq. In *International Conference on Interactive Theorem Proving*. LIPIcs.

Hostert, J., Koch, M., and Kirst, D. (2021). A toolbox for mechanised first-order logic. In *Coq Workshop*.

Kirst, D. (2018). Foundations of mathematics: A discussion of sets and types. Bachelor's thesis, Saarland University.

Kirst, D. and Hermes, M. (2021). Synthetic undecidability and incompleteness of first-order axiom systems in Coq. In *International Conference on Interactive Theorem Proving*. LIPIcs.

Kirst, D., Hostert, J., Dudenhefner, A., Forster, Y., Hermes, M., Koch, M., Larchey-Wendling, D., Mück, N., Peters, B., Smolka, G., and Wehr, D. (2022). A Coq library for mechanised first-order logic. In *Coq Workshop*.

Kirst, D. and Larchey-Wendling, D. (2022). Trakhtenbrot's theorem in Coq: Finite model theory through the constructive lens. *Logical Methods in Computer Science*, 18.

Kirst, D. and Peters, B. (2023). Gödel's theorem without tears: Essential incompleteness in synthetic computability. In *Annual conference of the European Association for Computer Science Logic*. LIPIcs.

Bibliography IV

Kirst, D. and Rech, F. (2021). The generalised continuum hypothesis implies the axiom of choice in Coq. In *International Conference on Certified Programs and Proofs*. ACM.

Kirst, D. and Smolka, G. (2019). Categoricity results and large model constructions for second-order ZF in dependent type theory. *Journal of Automated Reasoning*, 63(2):415–438.

Kleene, S. C. (1951). A symmetric form of Gödel's theorem. *Journal of Symbolic Logic*, 16(2).

Koch, M. and Kirst, D. (2022). Undecidability, incompleteness, and completeness of second-order logic in Coq. In *International Conference on Certified Programs and Proofs*. ACM.

Kreisel, G. (1962). On weak completeness of intuitionistic predicate logic. *The Journal of Symbolic Logic*, 27(2):139–158.

Krivine, J.-L. (1996). Une preuve formelle et intuitionniste du théorčme de complétude de la logique classique. *Bulletin of Symbolic Logic*, 2(4):405–421.

Larchey-Wendling, D. and Forster, Y. (2019). Hilbert's tenth problem in Coq. In *4th International Conference on Formal Structures for Computation and Deduction*, volume 131 of *LIPIcs*, pages 27:1–27:20.

O'Connor, R. (2005). Essential incompleteness of arithmetic verified by Coq. In *International Conference on Theorem Proving in Higher Order Logics*, pages 245–260. Springer.

Bibliography V

Paulin-Mohring, C. (1993). Inductive definitions in the system Coq - rules and properties. In *International Conference on Typed Lambda Calculi and Applications*, pages 328–345. Springer.

Paulson, L. C. (2015). A mechanised proof of Gödel's incompleteness theorems using Nominal Isabelle. *Journal of Automated Reasoning*, 55(1):1–37.

Popescu, A. and Traytel, D. (2021). Distilling the requirements of Gödel's incompleteness theorems with a proof assistant. *Journal of Automated Reasoning*, 65(7):1027–1070.

Richman, F. (1983). Church's thesis without tears. *The Journal of symbolic logic*, 48(3):797–803.

Rosser, B. (1936). Extensions of some theorems of Gödel and Church. *The journal of symbolic logic*, 1(3):87–91.

Scott, D. (2008). The algebraic interpretation of quantifiers: Intuitionistic and classical. In A. Ehrenfeucht, V. M. and Srebrny, M., editors, *Andrzej Mostowski and Foundational Studies*. IOS Press.

Shankar, N. (1986). *Proof-checking metamathematics*. The University of Texas at Austin. PhD Thesis.

Sierpiński, W. (1947). L'hypothèse généralisée du continu et l'axiome du choix. *Fundamenta Mathematicae*, 1(34):1–5.

Simpson, S. G. (2009). *Subsystems of second order arithmetic*, volume 1. Cambridge University Press.

Stark, K. (2019). Mechanising syntax with binders in Coq.

Bibliography VI

Stark, K., Schäfer, S., and Kaiser, J. (2019). Autosubst 2: reasoning with multi-sorted de Bruijn terms and vector substitutions. In *International Conference on Certified Programs and Proofs*, pages 166–180. ACM.

Turing, A. M. (1937). On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London mathematical society*, 2(1):230–265.

Veldman, W. (1976). An intuitionistic completeness theorem for intuitionistic predicate logic. *The Journal of Symbolic Logic*, 41(1):159–166.

Werner, B. (1997). Sets in types, types in sets. In *International Symposium on Theoretical Aspects of Computer Software*, pages 530–546. Springer.

Zermelo, E. (1930). Über Grenzzahlen und Mengenbereiche: Neue Untersuchungen über die Grundlagen der Mengenlehre. *Fundamenta Mathematicæ*, 16:29–47.

Synthetic Decidability and Enumerability

A problem interpreted as a predicate $p : X \to \mathbb{P}$ on a type X is

decidable if there is a function $f : X \to \mathbb{B}$ with

$$\forall x.\, p\, x \leftrightarrow f\, x = \mathrm{tt},$$

enumerable if there is a function $f : \mathbb{N} \to \mathbb{O}(X)$ with

$$\forall x.\, p\, x \leftrightarrow \exists n.\, f\, n = \ulcorner x \urcorner.$$

Fact

Let $p : X \to \mathbb{P}$ be a predicate, then p is

- *decidable iff $\forall x.\, p\, x + \neg(p\, x)$ is inhabited and*
- *enumerable iff there is $L : \mathbb{N} \to \mathbb{L}(X)$ s.t. $\forall x.\, p\, x \leftrightarrow \exists n.\, x \in L\, n$.*

Synthetic Many-One Reductions

Given predicates $p : X \to \mathbb{P}$ and $q : Y \to \mathbb{P}$ we call a function $f : X \to Y$ a (many-one) reduction from p to q if

$$\forall x.\, p\, x \leftrightarrow q\, (f\, x).$$

We write $p \preccurlyeq q$ if a reduction from p to q exists.

Theorem (Reduction)

Let p and q be predicates on data types with $p \preccurlyeq q$.

- *If q is decidable/enumerable/co-enumerable, then so is p.*
- *If p is not co-enumerable, then q is not co-enumerable.*

Proof.

If f witnesses $p \preccurlyeq q$ and g decides q, then $g \circ f$ decides p. □

Framework: Comparison

Development	Signature	Binding	(AI)-Rule	Weakening
O'Connor	arbitrary	named	side-condition	n.a.
Ilik	monadic	locally-nameless	co-finite	easy
Herbelin et al.	dyadic	locally-named	side-condition	needs renaming
Han and van Doorn	arbitrary	de Bruijn	shifting	easy
Laurent	full	anti-loc.-namel.	shifting	easy
Our framework	arbitrary	de Bruijn	shifting	easy

Framework: Tool Support

Tools developed by Hostert, Koch, and Kirst (2021):

- HOAS-input language
  - Concrete formulas can be written with Coq binders instead of de Bruijn indices
  - Eases interaction with the syntax
- Proof mode (inspired by Iris proof mode)
  - Tactic and notation layer hiding the proof rules
  - Eases interaction with the deduction systems
- Reification tactic (employing MetaCoq)
  - Extracts first-order formulas from Coq predicates
  - Eases interaction with the semantics

Framework: Usability (Proof Mode)

```
    frewrite (ax_add_zero y).
    fapply ax_refl.
  - fintros "x" "IH" "y".
    frewrite (ax_add_rec (σ y) x).
    frewrite ("IH" y).
    frewrite (ax_add_rec y x). fapply ax_refl.
Qed.

Lemma add_comm :
  FAI ⊢ << ∀' x y, x ⊕ y == y ⊕ x.
Proof.
  fstart. fapply ((ax_induction (<< Free x, ∀' y, x ⊕ y == y ⊕ x))).
  - fintros.
    frewrite (ax_add_zero x).
    frewrite (add_zero_r x).
    fapply ax_refl.
  - fintros "x" "IH" "y".
    frewrite (add_succ_r y x).
    frewrite <- ("IH" y).
    frewrite (ax_add_rec y x).
    fapply ax_refl.
Qed.

Lemma pa_eq_dec :
  FAI ⊢ << ∀' x y, (x == y) ∨ ¬ (x == y).
Proof.
  fstart.
  fapply ((ax_induction (<< Free x, ∀' y, (x == y) ∨ ¬ (x == y)))).
```

```
1 goal
p : peirce
x, y : term
______________________________________(1/1)
 FAI
"IH" : ∀  x0, x`[↑] ⊕  x0 ==  x0 ⊕ x`[↑]
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
(σ x ⊕ y == y ⊕ σ x)
```

Messages ↗ Errors ↗ Jobs ↗

https://github.com/dominik-kirst/coq-library-undecidability/blob/fol-library/theories/FOL/Proofmode/DemoPA.v

Framework: Usability (Reification Tactic)

```
Proof.
elim a using PA_induction.
- represent.
- eapply ieq_trans. 1:apply (add_zero_l (iS b)).
  apply ieq_congr_succ, ieq_sym, add_zero_l.
- intros d IH.
  eapply ieq_trans. 1:apply (add_succ_l d (iS b)).
  apply ieq_congr_succ. eapply ieq_trans.
  + apply IH.
  + apply ieq_sym, add_succ_l.
Qed.

Lemma add_comm a b : a i⊕ b i= b i⊕ a.
Proof.
elim a using PA_induction.
- represent.
- eapply ieq_trans.
  + apply (add_zero_l b).
  + apply ieq_sym, (add_zero_r b).
- intros a' IH.
  eapply ieq_trans. 2:eapply ieq_trans.
  + apply (add_succ_l a' b).
  + apply ieq_congr_succ, IH.
  + apply ieq_sym, add_succ_r.
Qed.
```

```
1 goal
D' : Type
I : interp D'
D_fulfills : forall (f : form) (rho : env D'),
             PAeq f -> rho ⊨ f
a, b : D'
______________________________________(1/1)
representableP 1 (fun a0 : D => a0 i⊕ b i= b i⊕ a0)
```

Messages ↗ Errors ↗ Jobs ↗

https://github.com/dominik-kirst/coq-library-undecidability/blob/fol-library/theories/FOL/Reification/DemoPA.v

Framework: Deduction Systems

Proof rules are represented as inductive predicates relating a context Γ to a formula φ:

$$\cdots$$

$$\frac{\Gamma[\uparrow] \vdash \varphi}{\Gamma \vdash \forall\varphi} \qquad \frac{\Gamma \vdash \forall\varphi}{\Gamma \vdash \varphi[t]} \qquad \frac{\Gamma \vdash \varphi[t]}{\Gamma \vdash \exists\varphi} \qquad \frac{\Gamma \vdash \exists\varphi \quad \Gamma[\uparrow], \varphi \vdash \psi[\uparrow]}{\Gamma \vdash \psi}$$

$$\cdots$$

- Quantifier rules use shifted contexts $\Gamma[\uparrow]$ so that x_0 acts as canonical free variable
- Trivialises structural properties like substitutivity and weakening
- Availability of classical rules regulated via type class flag
- Similar representation of sequent calculi and other systems

Framework: Semantics

Tarski models $\mathcal{M}$ are represented as a domain type D and symbol interpretations:

$$f^{\mathcal{M}} \;:\; D^{|f|} \to D \qquad\qquad P^{\mathcal{M}} \;:\; D^{|P|} \to \mathbb{P}$$

- Interpretation of terms and formulas based on assignments $\rho : \mathbb{N} \to D$
- Term evaluation $\hat{\rho}\, t$ defined recursively, main rule $\hat{\rho}\,(f\,\vec{t}\,) \;:=\; f^{\mathcal{M}}\,(\hat{\rho}\,\vec{t}\,)$
- Formula satisfaction $\rho \vDash \varphi$ defined recursively, main rule $\rho \vDash P\,\vec{t} \;:=\; P^{\mathcal{M}}\,(\hat{\rho}\,\vec{t}\,)$
- Induces the logical entailment relation $\Gamma \vDash \varphi$

Framework: Axiom Systems

Concrete axiom systems $\mathcal{A}$ are modelled as predicates of formulas over a specific signature.

For the example of Peano arithmetic (PA), we instantiate to the arithmetical signature

$$(O,\, S\,\_\,,\ \_ + \_\,,\ \_ \times \_\,;\ \_ \equiv \_)$$

and collect the usual axioms, with the induction scheme represented as all instances of

$$\varphi[O] \to (\forall x.\, \varphi[x] \to \varphi[S\,x]) \to \forall x.\, \varphi[x].$$

- Include fragments of PA like Robinson's Q, also several variants of ZF set theory
- Equality $\equiv$ seen as axiomatised symbol of the signature rather than a logical primitive
- Axiom systems $\mathcal{A}$ induce relatives deductive and semantic theories $\mathcal{A} \vdash \varphi$ and $\mathcal{A} \vDash \varphi$

Framework: Syntax (Coq)

```
Context {sig_funcs : funcs_signature}.

Inductive term  : Type :=
  | var : nat -> term
  | func : forall (f : syms), vec term (ar_syms f) -> term.

Context {sig_preds : preds_signature}.

Inductive falsity_flag := falsity_off | falsity_on.
Existing Class falsity_flag.

Class operators := {binop : Type ; quantop : Type}.
Context {ops : operators}.

Inductive form : falsity_flag -> Type :=
  | falsity : form falsity_on
  | atom {b} : forall (P : preds), vec term (ar_preds P) -> form b
  | bin {b} : binop -> form b -> form b -> form b
  | quant {b} : quantop -> form b -> form b.
```

Framework: Deduction Systems (Coq)

```
Context {sig_funcs : funcs_signature}.
Context {sig_preds : preds_signature}.

Reserved Notation 'A ⊢ phi' (at level 61).

Inductive peirce := class | intu.
Existing Class peirce.

Inductive prv : forall (ff : falsity_flag) (p : peirce), list form -> form -> Prop :=
  | II {ff} {p} A phi psi : phi::A ⊢ psi -> A ⊢ phi --> psi
  | IE {ff} {p} A phi psi : A ⊢ phi --> psi -> A ⊢ phi -> A ⊢ psi
  | AllI {ff} {p} A phi : map (subst_form ↑) A ⊢ phi -> A ⊢ ∀ phi
  | AllE {ff} {p} A t phi : A ⊢ ∀ phi -> A ⊢ phi[t..]
  | Exp {p} A phi : prv p A falsity -> prv p A phi
  | Ctx {ff} {p} A phi : phi el A -> A ⊢ phi
  | Pc {ff} A phi psi : prv class A (((phi --> psi) --> phi) --> phi)
 where 'A ⊢ phi' := (prv _ A phi).
```

Framework: Semantics (Coq)

```
Context {domain : Type}.

Class interp := B_I
  { i_func : forall f : syms, vec domain (ar_syms f) -> domain ;
    i_atom : forall P : preds, vec domain (ar_preds P) -> Prop ; }.

Definition env := nat -> domain.

Context {I : interp}.

Fixpoint eval (rho : env) (t : term) : domain := match t with
  | var s => rho s
  | func f v => i_func (Vector.map (eval rho) v) end.

Fixpoint sat {ff : falsity_flag} (rho : env) (phi : form) : Prop := match phi with
  | atom P v => i_atom (Vector.map (eval rho) v)
  | falsity => False
  | bin Impl phi psi => sat rho phi -> sat rho psi
  | quant All phi => forall d : domain, sat (d .: rho) phi end.
```

Analysing Completeness Theorems in Constructive Meta-Theory

Confusing situation in the literature on first-order logic:

- Completeness equivalent to Boolean Prime Ideal Theorem (Henkin, 1954)
- Completeness requires Markov's Principle (Kreisel, 1962)
- Completeness equivalent to Weak König's Lemma (Simpson, 2009)
- Completeness holds fully constructively (Krivine, 1996)

Systematic investigation missing:

- Started consolidation by Herbelin and Ilik (2016) and Forster et al. (2021)
- Comprehensive overview of current landscape by Herbelin (2022)

The Issue with Disjunction

Truth Lemma case for disjunctions $\varphi \lor \psi$:

$$\begin{aligned} \varphi \lor \psi \in \mathcal{T} &\overset{?}{\Longleftrightarrow} \mathcal{T} \Vdash \varphi \lor \psi \\ &\overset{def}{\Longleftrightarrow} \mathcal{T} \Vdash \varphi \lor \mathcal{T} \Vdash \psi \\ &\overset{IH}{\Longleftrightarrow} \varphi \in \mathcal{T} \lor \psi \in \mathcal{T} \end{aligned}$$

- So we really need prime theories for disjunctions
- Primeness from Lindenbaum Extension is constructive no-go

Backwards Analysis

Two proofs of Quasi-Completeness from incomparable principles...

Fact

Model Existence implies WLEM.

Proof.

Given P, use model existence on $\mathcal{T} := \{x_0 \vee \neg x_0\} \cup \{x_0 \mid P\} \cup \{\neg x_0 \mid \neg P\}$. We have $\mathcal{T} \not\vdash \bot$ so if $\mathcal{M} \Vdash \mathcal{T}$, then either $\mathcal{M} \Vdash x_0$ or $\mathcal{M} \Vdash \neg x_0$, so either $\neg\neg P$ or $\neg P$, respectively. □

Fact

Quasi-Completeness implies the following principle: $\forall p : \mathbb{N} \to \mathbb{P}. \neg\neg(\forall n. \neg p\, n \vee \neg\neg p\, n)$

Proof.

Using similar tricks for $\mathcal{T} := \{x_n \vee \neg x_n\} \cup \{x_n \mid p\, n\} \cup \{\neg x_n \mid \neg p\, n\}$, see backup slide. □

Obvious consequence both from WLEM and DNS, maybe enough for Quasi-Completeness?

Weak Double-Negation Shift (Preliminary Name)

$$\mathsf{WDNS} \;:=\; \forall p : \mathbb{N} \to \mathbb{P}.\, \neg\neg(\forall n.\, \neg p\, n \vee \neg\neg p\, n)$$

Lemma

Assuming WDNS, every stable quasi-prime theory is not not prime.

Proof.

Assume $\mathcal{T}$ not prime and derive a contradiction. Given the negative goal, from WDNS we obtain $\forall \varphi.\, \neg(\varphi \in \mathcal{T}) \vee \neg\neg(\varphi \in \mathcal{T})$. This yields exactly the instances of WLEM needed to derive that $\mathcal{T}$ is prime, contradiction. □

WDNS turns stable predicates $p : \mathbb{N} \to \mathbb{P}$ not not decidable, contributes to Fan Theorem

Already the Lemma turns out to be enough for Quasi-Completeness!

Quasi-Completeness via WDNS

Refined proof outline using WDNS:

- Lindenbaum Extension: if $\mathcal{T} \nvdash \varphi$ then there is stable not not prime $\mathcal{T}'$ with $\mathcal{T}' \nvdash \varphi$
- Universal Model: consistent stable prime theories related by inclusion
- Truth Lemma: $\varphi \in \mathcal{T} \iff \mathcal{T} \Vdash \varphi$
- Pseudo Model Existence: if $\mathcal{T} \nvdash \varphi$ then there not not is $\mathcal{M}$ with $\mathcal{M} \Vdash \mathcal{T}$ and $\mathcal{M} \nVdash \varphi$
- Quasi-Completeness: if $\mathcal{T} \Vdash \varphi$ then $\neg\neg(\mathcal{T} \vdash \varphi)$
- Completeness: anyway no constructive consequence of Quasi-Completeness

Encoding the Post Correspondence Problem

We use the signature $\Sigma_{\mathrm{PCP}} := (\{\star^0, e^0, f^1_{\mathrm{tt}}, f^1_{\mathrm{ff}}\}; \{P^2, \prec^2, \equiv^2\})$:

- Chains like $f_{\mathrm{ff}}(f_{\mathrm{tt}}(e))$ represent strings while $\star$ signals overflow
- P concerns only defined values and $\prec$ is a strict ordering:

$$\begin{aligned}\varphi_P &:= \dot\forall xy.\, P\,x\,y \dot\rightarrow x \not\equiv \star \,\dot\wedge\, y \not\equiv \star\\ \varphi_\prec &:= (\dot\forall x.\, x \not\prec x) \,\dot\wedge\, (\dot\forall xyz.\, x \prec y \dot\rightarrow y \prec z \dot\rightarrow x \prec z)\end{aligned}$$

- Sanity checks on f regarding overflow, disjointness, and injectivity:

$$\varphi_f := \begin{pmatrix} f_{\mathrm{tt}}\,\star \equiv \star \,\dot\wedge\, f_{\mathrm{ff}}\,\star \equiv \star \\ \dot\forall x.\, f_{\mathrm{tt}}\,x \not\equiv e \\ \dot\forall x.\, f_{\mathrm{ff}}\,x \not\equiv e \end{pmatrix} \dot\wedge \begin{pmatrix} \dot\forall xy.\, f_{\mathrm{tt}}\,x \not\equiv \star \dot\rightarrow f_{\mathrm{tt}}\,x \equiv f_{\mathrm{tt}}\,y \dot\rightarrow x \equiv y \\ \dot\forall xy.\, f_{\mathrm{ff}}\,x \not\equiv \star \dot\rightarrow f_{\mathrm{ff}}\,x \equiv f_{\mathrm{ff}}\,y \dot\rightarrow x \equiv y \\ \dot\forall xy.\, f_{\mathrm{tt}}\,x \equiv f_{\mathrm{ff}}\,y \dot\rightarrow f_{\mathrm{tt}}\,x \equiv \star \,\dot\wedge\, f_{\mathrm{ff}}\,y \equiv \star \end{pmatrix}$$

Trakhtenbrot's Theorem

Given an instance R of PCP, we construct a formula φ_R by:

$$\varphi_R := \varphi_P \mathbin{\dot\land} \varphi_\prec \mathbin{\dot\land} \varphi_f \mathbin{\dot\land} \varphi_\triangleright \mathbin{\dot\land} \dot\exists x.\, P\, x\, x$$

Crucially, we enforce that P satisfies the inversion principle of $R \triangleright (s,t)$:

$$\varphi_\triangleright := \dot\forall xy.\, P\, x\, y \mathbin{\dot\to} \dot{\bigvee_{(s,t)\in R}} \mathbin{\dot\lor} \begin{cases} x \equiv \overline{s} \mathbin{\dot\land} y \equiv \overline{t} \\ \dot\exists uv.\, P\, u\, v \mathbin{\dot\land} x \equiv \overline{s}u \mathbin{\dot\land} y \equiv \overline{t}v \mathbin{\dot\land} (u,v) \prec (x,y) \end{cases}$$

Theorem

PCP R *iff* $\mathrm{FSATEQ}(\Sigma_{\mathrm{PCP}};\equiv)\varphi_R$, *hence* $\mathrm{PCP} \preccurlyeq \mathrm{FSATEQ}(\Sigma_{\mathrm{PCP}};\equiv)$.

Proof.

If R has a solution of length n, then φ_R is satisfied by the model of strings of length bounded by n. Conversely, if $\mathcal{M} \vDash_\rho \varphi_R$ we can extract a solution of R from $\varphi_\triangleright$ by well-founded induction on $\prec^{\mathcal{M}}$ (which is applicable since $\mathcal{M}$ is finite). □

Sketch for Peano Arithmetic

Use axiomatisation PA over standard signature $(0, \mathsf{S}, +, \cdot\,; \equiv)$.

Diophantine constraints (cf. Larchey-Wendling and Forster (2019)):

- Instances are lists L of constraints $x_i = 1 \mid x_i + x_j = x_k \mid x_i \cdot x_j = x_k$
- L is solvable if there is an evaluation $\eta : \mathbb{N} \to \mathbb{N}$ solving all constraints

Theorem

$L = [c_1, \dots, c_k]$ *with maximal index* x_n *is solvable iff* $\mathsf{PA} \vDash \exists^n c_1 \land \dots \land c_k$.

Proof.

If L has solution η instantiate the existential quantifiers with numerals $\overline{\eta_1}, \dots, \overline{\eta_n}$. Then the axioms of PA entail the constraints.

If $\mathsf{PA} \vDash \exists^n c_1 \land \dots \land c_k$ use the standard model $\mathbb{N}$ to extract solution η. □

Sketch for ZF Set Theory

Use axiomatisation ZF over explicit signature $(\emptyset, \{\_,\_\}, \bigcup, \mathcal{P}, \omega\,; \equiv, \in)$.

Reduction from PCP:

- Boolean encoding: $\overline{\mathsf{tt}} = \{\emptyset, \emptyset\}$ and $\overline{\mathsf{tt}} = \emptyset$
- String encoding: $\overline{\mathsf{tt}\,\mathsf{ff}\,\mathsf{ff}\,\mathsf{tt}} = (\overline{\mathsf{tt}}, (\overline{\mathsf{ff}}, (\overline{\mathsf{tt}}, (\overline{\mathsf{ff}}, \emptyset))))$
- Stack encoding: $\overline{S} = \{(\overline{s_1}, \overline{t_1}), \dots, (\overline{s_k}, \overline{t_k})\}$
- Combination encoding: $S +\!\!+ B := \bigcup_{s/t \in S} \{(\overline{s}x, \overline{t}y) \mid (x, y) \in B\}$
- $f \rhd n := (\emptyset, \overline{S}) \in f \land \forall (k, B) \in f.\, k \in n \rightarrow (k+1, S +\!\!+ B) \in f$

$$\varphi_S := \exists f, n, B, x.\, n \in \omega \land f \rhd n \land (n, B) \in f \land (x, x) \in B$$

Theorem

PCP S *iff* ZF $\vDash \varphi_S$ *and* PCP S *iff* ZF $\vdash \varphi_S$.

Proof.

Direction $\rightarrow$ by internal proofs and $\leftarrow$ relies on standard model $\mathcal{S}$. □

Incompleteness: Halting Problem

Fact

K_Θ *is undecidable, in fact for every candidate decider* $d : \mathbb{N} \rightharpoonup \mathbb{B}$ *with*

$$\forall x.\, \mathsf{K}_\Theta\, x \leftrightarrow d\, x \downarrow \mathsf{tt}$$

one can construct a concrete value x *with* $\neg \mathsf{K}_\Theta\, x$ *such that* $d\, x \uparrow$.

Proof.

We first define the partial function $f : \mathbb{N} \rightharpoonup \mathbb{B}$ such that $f\, x \downarrow \mathsf{tt}$ whenever $d\, x \downarrow \mathsf{ff}$ and $f\, x \uparrow$ otherwise. Now using EPF we obtain a code c for f and deduce for $x := c$ that

$$d\, x \downarrow \mathsf{tt} \;\Leftrightarrow\; \mathsf{K}_\Theta\, x \;\Leftrightarrow\; \Theta_x\, x \downarrow \;\Leftrightarrow\; f\, x \downarrow \;\Leftrightarrow\; f\, x \downarrow \mathsf{tt} \;\Leftrightarrow\; d\, c \downarrow \mathsf{ff}$$

from which we conclude $d\, x \uparrow$. That K_Θ is not decidable follows since every decider $\mathbb{N} \to \mathbb{B}$ would induce a total candidate decider $\mathbb{N} \rightharpoonup \mathbb{B}$. □

Incompleteness: Recursive Inseparability

Fact

K^1_Θ and K^0_Θ are recursively inseparable, in fact for every candidate separator $s : \mathbb{N} \rightharpoonup \mathbb{B}$ with

$$\forall x.\ (\mathsf{K}^1_\Theta\, x \to s\, x \downarrow \mathsf{tt}) \ \wedge\ (\mathsf{K}^0_\Theta\, x \to s\, x \downarrow \mathsf{ff})$$

one can construct a concrete value x with $\neg\mathsf{K}^1_\Theta\, x$ and $\neg\mathsf{K}^0_\Theta\, x$ such that $s\, x \uparrow$.

Proof.

We define the partial function $f : \mathbb{N} \rightharpoonup \mathbb{B}$ such that $f\, x \downarrow \mathsf{ff}$ if $s\, x \downarrow \mathsf{tt}$, $f\, x \downarrow \mathsf{tt}$ if $s\, x \downarrow \mathsf{ff}$, and $f\, x \uparrow$ otherwise. Using EPF we obtain a code c for f and deduce for $x := c$ that

$$\begin{aligned} s\, x \downarrow \mathsf{tt} \ &\Leftrightarrow\ f\, x \downarrow \mathsf{ff} \ \Leftrightarrow\ \Theta_x\, x \downarrow 0 \ \Leftrightarrow\ \mathsf{K}^0_\Theta\, x \ \Rightarrow\ s\, x \downarrow \mathsf{ff} \\ s\, x \downarrow \mathsf{ff} \ &\Leftrightarrow\ f\, x \downarrow \mathsf{tt} \ \Leftrightarrow\ \Theta_x\, x \downarrow 1 \ \Leftrightarrow\ \mathsf{K}^1_\Theta\, x \ \Rightarrow\ s\, x \downarrow \mathsf{tt} \end{aligned}$$

from which we conclude $s\, x \uparrow$. □

Incompleteness: Partial Decider

Lemma (Partial Decider)

One can construct a partial function $d_{\mathcal{S}} : \mathbb{S} \rightharpoonup \mathbb{B}$ *with:*

$$\forall \varphi.\ (\vdash \varphi \leftrightarrow d_{\mathcal{S}}\, \varphi \downarrow \mathsf{tt}) \ \land\ (\vdash \dot{\neg} \varphi \leftrightarrow d_{\mathcal{S}}\, \varphi \downarrow \mathsf{ff})$$

Note that by this specification $d_{\mathcal{S}}$ *exactly diverges on the independent sentences of* $\mathcal{S}$.

Lemma

Let $d_{\mathcal{S}}$ *be the partial decider to* $\mathcal{S}$.

1. *If* $\mathcal{S}$ *represents* K_{Θ}, *then* $d_{\mathcal{S}}$ *is a candidate decider for* K_{Θ}.
2. *If* $\mathcal{S}$ *separates* K^1_{Θ} *and* K^0_{Θ}, *then* $d_{\mathcal{S}}$ *is a candidate separator for* K^1_{Θ} *and* K^0_{Θ}.

First-Order Set Theory[6]

Axiomatise ZF set theory over a suitable signature using first-order formulas:

$$\forall xy.\, x \subseteq y \to y \subseteq x \to x = y \qquad \forall x.\, x \notin \emptyset \qquad \forall xy.\, y \in \mathcal{P}(x) \leftrightarrow y \subseteq x$$

Separation and replacement represented as axiom schemes in $\varphi(x)$ and functional $\psi(x, y)$:

$$\forall x.\, \exists y.\, \forall z.\, z \in y \leftrightarrow z \in x \land \varphi(z) \qquad \forall x.\, \exists y.\, \forall z.\, z \in y \leftrightarrow \exists u \in x.\, \psi(u, z)$$

Verifying a set-theoretic result means to derive $\mathrm{ZF} \vdash \varphi$ or $\mathrm{ZF} \vDash \varphi$

- Limited to first-order encodings of functions, ordinals, cumulative hierarchy, etc.
- Undecidability obtained with usual method, provided a model exists (cf. Werner (1997))
- Incompleteness applies both to deduction and semantics

[6] Kirst and Hermes (2021)

Second-Order Set Theory[7]

Axiomatise a type $\mathcal{M}$ with relation $\in : \mathcal{M} \to \mathcal{M} \to \mathbb{P}$ and set-theoretic operations:

$$\forall xy : \mathcal{M}.\, x \subseteq y \to y \subseteq x \to x = y \qquad \forall x : \mathcal{M}.\, x \notin \emptyset \qquad \forall xy : \mathcal{M}.\, y \in \mathcal{P}(x) \leftrightarrow y \subseteq x$$

Separation and replacement quantify over all predicates, as intended by Zermelo (1930):

$$\forall p : \mathcal{M} \to \mathbb{P}.\, \forall xy.\, y \in p \cap x \leftrightarrow y \in x \land p\, x \qquad \forall F : \mathcal{M} \to \mathcal{M}.\, \forall xy.\, y \in F@x \leftrightarrow \exists z \in x.\, y = F\, z$$

Verifying a set-theoretic result means to show it for $\mathcal{M}$ (possibly assuming UC, FE, PE)

- Function spaces coincide, ordinals and cumulative hierarchy can be described inductively
- Undecidability could be shown if given in second-order syntax (e.g. Koch and Kirst, 2022)
- Incompleteness applies only to deduction, semantics is nearly determined

[7] Kirst (2018)

Synthetic Set Theory[8]

Use type-theoretic structure to represent set-theoretic operations:

$$\mathbb{0}, \mathbb{B}, \mathbb{N}, X \times Y, X + Y, X \to Y, X \to \mathbb{P}, \ldots$$

Separation is a sigma type over a predicate, replacement a sigma type over a function range:

$$\lambda X.\, \lambda p : X \to \mathbb{P}.\, \Sigma x : X.\, p\, x \qquad \lambda XY.\, \lambda F : X \to Y.\, \Sigma y : Y.\, \exists x : X.\, y = F\, x$$

Verifying a set-theoretic result means to show a type-theoretic result (assuming UC, FE, PE)

- No intermediate axiomatisation at all, simply work with type-theoretic primitives
- No external results like undecidability or incompleteness can be shown
- Internal results may may rely on alternative constructions

[8] Kirst and Rech (2021)

Constructing Large Ordinals: $|\aleph(A)| \not\leq |A|$

Definition

The Hartogs number of a set A is the class $\aleph(A) := \lambda\alpha \in \mathcal{O}.\, |\alpha| \leq |A|$.

Theorem

The Hartogs number $\aleph(A)$ of A satisfies the following properties:

1. $|\aleph(A)| \leq |\mathcal{P}^6(A)|$
2. $\aleph(A) \in \mathcal{O}$
3. $|\aleph(A)| \not\leq |A|$

Proof.

1. By representing ordinals $|\alpha| \leq |A|$ as well-ordered subsets of A.
2. Straightforward by definition of ordinals.
3. Straightforward by definition of $\aleph(A)$. □

Sierpiński's Theorem: Proof

Proof.

Assume GCH, to show AC it suffices to show that every infinite type is well-orderable.
So for some infinite X, apply GCH to the situation obtained by Lemma 1:

$$|\mathcal{P}^2(X)| \leq |\mathcal{P}^2(X) + \aleph(X)| \leq |\mathcal{P}^3(X)|$$

- $|\mathcal{P}^2(X) + \aleph(X)| \leq |\mathcal{P}^2(X)|$ yields $|\aleph(X)| \leq |\mathcal{P}^2(X)|$, start again
- $|\mathcal{P}^3(X)| \leq |\mathcal{P}^2(X) + \aleph(X)|$ yields $|\mathcal{P}^3(X)| \leq |\aleph(X)|$ by Lemma 2 □

Lemma 1.

If X is infinite, then $|X| = |\mathbb{1} + X|$ and $|\mathcal{P}(X)| = |\mathcal{P}(X) + \mathcal{P}(X)|$. □

Lemma 2.

If $|\mathcal{P}(X)| \leq |X + Y|$ and $|X + X| \leq |X|$, then already $|\mathcal{P}(X)| \leq |Y|$. □