Synthetic Undecidability Proofs in Coq

Entscheidungsproblem, Trakhtenbrot's Theorem, First-Order Axiom Systems

Dominik Kirst

CS Theory Seminar (TSEM)
November 12th, 2020

SAARLAND UNIVERSITY
COMPUTER SCIENCE
SIC Saarland Informatics Campus

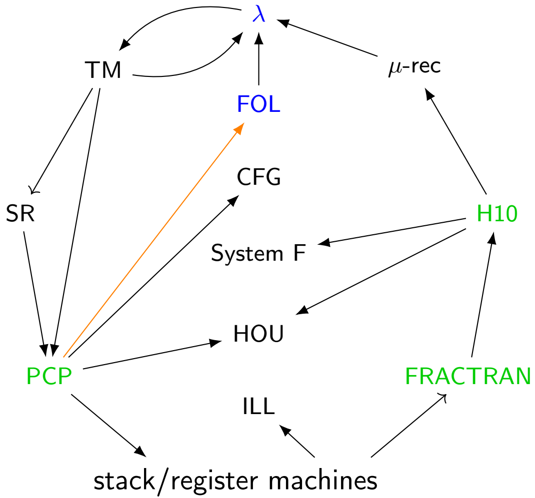
Our Coq Library of Undecidability Proofs*

- Merge of a few initial Coq developments:
  - Computablity theory using a cbv. lambda calculus
  - Synthetic computability
  - Initial undecidability proofs
- Extended with further undecidability reductions over past 2 years
- Unified framework to ease external contribution
- 9+ contributors and more than 100k lines of code
- 12+ related publications (ITP, CPP, IJCAR, FSCD, etc.)
- Currently roughly 13 (groups of) undecidable problems

---

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

Library Overview (Forster et al. (2020b))



- Classification in seed problems and target problems
- This talk: mostly the PCP → FOL edge

Outline

- Framework: Synthetic Undecidability
- Example 1: The Entscheidungsproblem
- Example 2: Trakhtenbrot's Theorem
- Example 3: First-Order Axiom Systems
- Conclusion

Framework: Synthetic Undecidability*

*Yannick Forster, K., and Gert Smolka at CPP'19.

How to mechanise decidability?

Conventional approach:

- Pick a concrete model of computation
  (Turing machines, μ-recursive functions, untyped λ-calculus, etc.)
- Invent a decision procedure for the given problem
- Explicitly code the algorithm in the chosen model!

Synthetic approach (Bauer (2006); Richman (1983)):

- Work in a constructive foundation, e.g. constructive type theory
- Define a decision procedure e.g. as a Boolean function
- Definable functions are computable, so that's it!

(Similar for other notions like enumerability and reducibility)

How to mechanise undecidability?

Problem of the synthetic approach:

- Constructive type theories like MLTT or CIC are consistent with classical assumption, rendering every problem decidable
- Proving a given problem undecidable is not outright possible

Possible solutions:

- Resort to a concrete model of computation
- Verify a synthetic reduction from an undecidable problem
  - Computability axioms could be used to obtain expected results

(Again similar for other negative notions of computability theory)

Coq's Type Theory

Main features of Coq's underlying CIC:

- Standard 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 $\mathcal{L}(X)$, options $\mathcal{O}(X)$, vectors X^n, ...
- Propositional universe $\mathbb{P}$ with logical connectives: $\to$, $\land$, $\lor$, $\forall$, $\exists$
- $\mathbb{P}$ is impredicative and separate from computational types

All definable functions $\mathbb{N} \to \mathbb{N}$ are computable!

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

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

enumerable if there is a function $f : \mathbb{N} \to \mathcal{O}(X)$ 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 $L : \mathbb{N} \to \mathcal{L}(X)$ s.t. $\forall x.\, p\,x \leftrightarrow \exists n.\, x \in L\,n$.*

Data Types

Computability theory is usually developed on computational domains.

A type X is called

- enumerable if $\lambda x.\top$ is enumerable,
- discrete if $\lambda xy.\, x = y$ is decidable, and
- data type if it is both enumerable and discrete.

Fact

Decidable predicates on data types are enumerable and co-enumerable.

Proof.

Let $f_X : \mathbb{N} \to \mathcal{O}(X)$ enumerate X and $f_p : X \to \mathbb{B}$ decide p. Then

$$f\ n := \text{match } f_X\ n \text{ with } \ulcorner x \urcorner \Rightarrow \text{if } f_p\, x \text{ then } \ulcorner x \urcorner \text{ else } \emptyset \mid \emptyset \Rightarrow \emptyset$$

defines an enumerator for p. □

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. □

The Post Correspondence Problem

Intuition: given a stack S of cards s/t, find a derivable match.

This (undecidable) problem can be expressed by an inductive predicate:

$$\frac{s/t \in S}{S \triangleright s/t} \qquad \frac{S \triangleright u/v \quad s/t \in S}{S \triangleright su/tv} \qquad \frac{S \triangleright s/s}{\mathsf{PCP}\ S}$$

Fact

The type $\mathbb{S}$ of stacks is a data type and PCP *is enumerable.*

Proof.

The former follows from closure properties and for the latter

$$\begin{aligned} L\,0 &:= [] \\ L\,(\mathsf{S}\,n) &:= L\,n \mathbin{+\!\!+} [(S,(s,t)) \mid S \in L_{\mathbb{S}}\,n, (s,t) \in S] \\ &\quad \mathbin{+\!\!+} [(S,(su,tv)) \mid (S,(u,v)) \in L\,n, (s,t) \in S] \end{aligned}$$

defines a list enumerator for $\lambda Sst.\, S \triangleright s/t$. □

Example 1:
The Entscheidungsproblem*

*Yannick Forster, K., and Gert Smolka at CPP'19.

General Idea

Given a FOL formula φ, is φ valid in all models?

Conventional outline following Turing:

- Encode Turing machine M as formula φ_M over custom signature
- Verify that M halts if and only if φ_M holds in all models
- (Argue why the used signature could have been minimised)

Our outline:

- Follow the simpler proof given in Manna (2003) using PCP
- Also don't bother with signature minimisation yet...

Syntax and Tarski Semantics

Terms and formulas are defined for a fixed signature:

$$\tau : \text{Term} := x \mid a \mid e \mid f_{\text{tt}}\,\tau \mid f_{\text{ff}}\,\tau \qquad x, a : \mathbb{N}$$
$$\varphi, \psi : \text{Form} := \dot{\bot} \mid Q \mid P\,\tau_1\,\tau_2 \mid \varphi \dot{\to} \psi \mid \dot{\forall} x.\,\varphi$$

Formulas are interpreted in models $\mathcal{I} = (D, \eta, e^{\mathcal{I}}, f_{\text{tt}}^{\mathcal{I}}, f_{\text{ff}}^{\mathcal{I}}, Q^{\mathcal{I}}, P^{\mathcal{I}})$ given a variable environment $\rho : \mathbb{N} \to D$:

$$\begin{aligned} \mathcal{I} \vDash_\rho \dot{\bot} &:= \bot \\ \mathcal{I} \vDash_\rho Q &:= Q^{\mathcal{I}} \\ \mathcal{I} \vDash_\rho P\,\tau_1\,\tau_2 &:= P^{\mathcal{I}}\,(\hat{\rho}\,\tau_1)\,(\hat{\rho}\,\tau_2) \\ \mathcal{I} \vDash_\rho \varphi \dot{\to} \psi &:= \mathcal{I} \vDash_\rho \varphi \to \mathcal{I} \vDash_\rho \psi \\ \mathcal{I} \vDash_\rho \dot{\forall} x.\,\varphi &:= \forall d : D.\, \mathcal{I} \vDash_{\rho[x:=d]} \varphi \end{aligned}$$

A formula φ is valid if $\mathcal{I} \vDash_\rho \varphi$ for all $\mathcal{I}$ and ρ.

A Standard Model

Strings can be encoded as terms, e.g. $\overline{\mathsf{tt}\,\mathsf{ff}\,\mathsf{ff}\,\mathsf{tt}} = f_{\mathsf{tt}}\,(f_{\mathsf{ff}}\,(f_{\mathsf{ff}}\,(f_{\mathsf{tt}}\,(e))))$.

The standard model $\mathcal{B}$ over the type $\mathcal{L}(\mathbb{B})$ of Boolean strings captures exactly the cards derivable from a fixed stack S:

$$e^{\mathcal{B}} := [] \qquad\qquad Q^{\mathcal{B}} := \mathsf{PCP}\,S$$

$$f_b^{\mathcal{B}}\,s := b :: s \qquad\qquad P^{\mathcal{B}}\,s\,t := S \triangleright s/t.$$

Lemma

Let $\rho : \mathbb{N} \to \mathcal{L}(\mathbb{B})$ be an environment for the standard model $\mathcal{B}$. Then $\hat{\rho}\,\overline{s} = s$ and $\mathcal{B} \vDash_\rho P\,\tau_1\,\tau_2 \leftrightarrow S \triangleright \hat{\rho}\,\tau_1/\hat{\rho}\,\tau_2$.

Undecidability of Validity

We express the constructors of $S \triangleright s/t$ and PCP as formulas:

$$\varphi_1 := [P\,\overline{s}\,\overline{t} \mid s/t \in S\,]$$
$$\varphi_2 := [\,\dot\forall xy.\, P\,x\,y \dot\to P\,(\overline{s}x)\,(\overline{t}y) \mid s/t \in S\,]$$
$$\varphi_3 := \dot\forall x.\, P\,x\,x \dot\to Q$$
$$\varphi_S := \varphi_1 \dot\to \varphi_2 \dot\to \varphi_3 \dot\to Q$$

Theorem

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

Proof.

Let φ_S be valid, so in particular $\mathcal{B} \vDash \varphi_S$. Since $\mathcal{B}$ satisfies all of φ_1, φ_2, and φ_3 it follows that $\mathcal{B} \vDash Q$ and thus PCP S.

Now suppose that $S \triangleright s/s$ for some s and that some model $\mathcal{I}$ satisfies all of φ_1, φ_2, and φ_3. Then $\mathcal{I} \vDash P\,\overline{s}\,\overline{s}$ by φ_1 and φ_2, hence $\mathcal{I} \vDash Q$ by φ_3, and thus $\mathcal{I} \vDash \varphi_S$. □

Undecidability of Satisfiability

Disclaimer: validity does not directly reduce to (co-)satisfiability!

- If φ is valid, then certainly $\dot\neg\varphi$ is unsatisfiable
- However, the converse does not hold constructively

Fortunately, we can give a direct reduction from the complement of PCP:

Theorem

$\neg$PCP S *iff* $\dot\neg\varphi_S$ *is satisfiable, hence co-*PCP *reduces to satisfiability.*

Proof.

If $\neg$PCP S, then $\mathcal{B} \vDash \dot\neg\varphi_S$ since $\mathcal{B} \vDash \varphi_S$ would yield PCP S as before. Now suppose there are $\mathcal{I}$ and ρ with $\mathcal{I} \vDash_\rho \dot\neg\varphi_S$. Then assuming PCP S yields the contradiction that φ_S is valid. □

Interlude: Completeness Theorems for FOL

Completeness of deduction systems for FOL relies on Markov's principle:

$$\mathsf{MP} := \forall f : \mathbb{N} \to \mathbb{B}.\, \neg\neg(\exists n.\, f\, n = \mathsf{tt}) \to \exists n.\, f\, n = \mathsf{tt}$$

MP is independent but admissible in Coq's type theory*

Theorem (cf. Yannick Forster, K., and Dominik Wehr at LFCS'20.)

- $\mathcal{T} \vDash \varphi$ *implies* $\neg\neg(\mathcal{T} \vdash_c \varphi)$ *for all* $\mathcal{T} : \mathsf{Form} \to \mathbb{P}$ *and* $\varphi : \mathsf{Form}$
- *If* $\mathcal{T}$ *is enumerable, then* MP *is equivalent to the stability of* $\mathcal{T} \vdash_c \varphi$

$\Rightarrow$ *Completeness for enumerable* $\mathcal{T}$ *is equivalent to* MP *and admissible*

Possible strategies:

a) Verify a weak reduction from PCP integrating the double negation
b) Obtain a standard reduction by proving $A \vdash_c \varphi_S$ by hand (done so far)

*Coquand/Mannaa '17, Pédrot/Tabareau '18

Undecidability of Minimal Provability

We define a minimal natural deduction system inductively:

$$\frac{\varphi \in A}{A \vdash \varphi}\ A \qquad \frac{\varphi :: A \vdash \psi}{A \vdash \varphi \dot{\rightarrow} \psi}\ II \qquad \frac{A \vdash \varphi \dot{\rightarrow} \psi \quad A \vdash \varphi}{A \vdash \psi}\ IE$$

$$\frac{A \vdash \varphi_a^x \quad a \notin \mathcal{P}(\varphi) \cup \mathcal{P}(A)}{A \vdash \dot{\forall} x.\, \varphi}\ AI \qquad \frac{A \vdash \dot{\forall} x.\, \varphi \quad \mathcal{V}(\tau) = \emptyset}{A \vdash \varphi_\tau^x}\ AE$$

A formula φ is provable if $\vdash \varphi$.

Fact (Soundness)

$A \vdash \varphi$ implies $A \vDash \varphi$, so provable formulas are valid.

Theorem

- PCP S *iff φ_S is provable.* *(proving $\vdash \varphi_S$ by hand)*
- *Provability is enumerable.* *(by giving a list enumerator)*

Undecidability of Classical Provability

We extend the deduction system by a classical rule for falsity:

$$\frac{A \vdash_c \dot\neg\dot\neg\varphi}{A \vdash_c \varphi}\ DN$$

Unfortunately, this rule is not sound constructively!

As a remedy, we define a Gödel-Gentzen-Friedman translation φ^Q of formulas φ such that $A \vdash_c \varphi$ implies $A^Q \vdash \varphi^Q$.

Theorem

PCP S *iff* φ_S *is classically provable, hence* PCP *reduces to classical ND.*

Proof.

If PCP S then $\vdash \varphi_S$ by the previous theorem and hence $\vdash_c \varphi_S$. Conversely, let $\vdash_c \varphi_S$ and hence $\vdash \varphi_S^Q$. Then by soundness $\mathcal{B} \vDash \varphi_S^Q$ which still implies $\mathcal{B} \vDash Q$ and PCP S as before. □

Example 2:
Trakhtenbrot's Theorem*

*K. and Dominique Larchey-Wendling at IJCAR'20.

General idea

Given a FOL formula φ, is φ finitely satisfiable?

Textbook proofs by dual reduction from the halting problem:*

- Encode Turing machine M as formula φ_M over custom signature
- Verify that the models of φ_M correspond to the runs of M
- Conclude that M halts if and only if φ_M has a finite model

Our mechanisation:

- Illustrates that one can still use PCP for a simpler reduction
- Signature minimisations are constructive for finite models

*e.g. Libkin (2010); Börger et al. (1997)

First-Order Satisfiability over Signatures

Given a signature $\Sigma = (\mathcal{F}_\Sigma; \mathcal{P}_\Sigma)$, we represent terms and formulas by:

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

$$\varphi, \psi \,:\, \mathsf{Form}_\Sigma \;::=\; \dot{\bot} \mid P\,\vec{t} \mid \varphi \,\dot{\Box}\, \psi \mid \dot{\nabla}\varphi \qquad\qquad (P : \mathcal{P}_\Sigma,\; \vec{t} : \mathsf{Term}_\Sigma^{|P|})$$

A model $\mathcal{M}$ over a domain D is a pair of interpretation functions:

$$\_^{\mathcal{M}} \;:\; \forall f : \mathcal{F}_\Sigma.\, D^{|f|} \to D \qquad\qquad \_^{\mathcal{M}} \;:\; \forall P : \mathcal{P}_\Sigma.\, D^{|P|} \to \mathbb{P}$$

For assignments $\rho : \mathbb{N} \to D$ define evaluation $\hat{\rho}\, t$ and satisfaction $\mathcal{M} \vDash_\rho \varphi$:

$$\hat{\rho}\, x \;:=\; \rho x \qquad\qquad \hat{\rho}\,(f\,\vec{t}) \;:=\; f^{\mathcal{M}}\,(\hat{\rho}\,\vec{t})$$

$$\mathcal{M} \vDash_\rho \dot{\bot} \;:=\; \bot \qquad\qquad \mathcal{M} \vDash_\rho \varphi \,\dot{\Box}\, \psi \;:=\; \mathcal{M} \vDash_\rho \varphi \,\Box\, \mathcal{M} \vDash_\rho \psi$$

$$\mathcal{M} \vDash_\rho P\,\vec{t} \;:=\; P^{\mathcal{M}}\,(\hat{\rho}\,\vec{t}) \qquad\qquad \mathcal{M} \vDash_\rho \dot{\nabla}\varphi \;:=\; \nabla a : D.\, \mathcal{M} \vDash_{a\cdot\rho} \varphi$$

$\mathsf{SAT}(\Sigma)\,\varphi :=$ there are $\mathcal{M}$ and ρ such that $\mathcal{M} \vDash_\rho \varphi$

Finiteness in Constructive Type Theory

Definition

A type X is finite if there exists a list l_X with $x \in l_X$ for all $x : X$.

This seems to be a good compromise:

- Easy to establish and work with
- Does not enforce discreteness
- Enough to get expected properties:
  - Every strict order on a finite type is well-founded
  - Every finite decidable equivalence relation admits a quotient on $\mathbb{F}_n$

FSAT(Σ) φ if additionally D is finite and all $P^{\mathcal{M}}$ are decidable

FSATEQ(Σ; ≡) φ if $x \equiv^{\mathcal{M}} y \leftrightarrow x = y$ for all $x, y : D$ (hence discrete)

Encoding the Post Correspondence Problem

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

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

$$\varphi_P := \dot\forall xy.\, P\,x\,y \mathrel{\dot\to} x \not\equiv \star \mathrel{\dot\land} y \not\equiv \star$$
$$\varphi_\prec := (\dot\forall x.\, x \not\prec x) \mathrel{\dot\land} (\dot\forall xyz.\, x \prec y \mathrel{\dot\to} y \prec z \mathrel{\dot\to} x \prec z)$$

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

$$\varphi_f := \begin{pmatrix} f_{\mathsf{tt}}\,\star \equiv \star \mathrel{\dot\land} f_{\mathsf{ff}}\,\star \equiv \star \\ \dot\forall x.\, f_{\mathsf{tt}}\,x \not\equiv e \\ \dot\forall x.\, f_{\mathsf{ff}}\,x \not\equiv e \end{pmatrix} \mathrel{\dot\land} \begin{pmatrix} \dot\forall xy.\, f_{\mathsf{tt}}\,x \not\equiv \star \mathrel{\dot\to} f_{\mathsf{tt}}\,x \equiv f_{\mathsf{tt}}\,y \mathrel{\dot\to} x \equiv y \\ \dot\forall xy.\, f_{\mathsf{ff}}\,x \not\equiv \star \mathrel{\dot\to} f_{\mathsf{ff}}\,x \equiv f_{\mathsf{ff}}\,y \mathrel{\dot\to} x \equiv y \\ \dot\forall xy.\, f_{\mathsf{tt}}\,x \equiv f_{\mathsf{ff}}\,y \mathrel{\dot\to} f_{\mathsf{tt}}\,x \equiv \star \mathrel{\dot\land} f_{\mathsf{ff}}\,y \equiv \star \end{pmatrix}$$

Trakhtenbrot's Theorem

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

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

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

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

Theorem

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

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). □

Signature Transformations

Given a finite and discrete signature Σ with arities bounded by n, we have:

$$\mathsf{FSATEQ}(\Sigma;\equiv) \preccurlyeq \mathsf{FSAT}(\Sigma) \preccurlyeq \mathsf{FSAT}(\mathbb{0}; P^{n+2}) \preccurlyeq \mathsf{FSAT}(\mathbb{0}; \in^2)$$

First reduction: axiomatise that $\equiv$ is a congruence for the symbols in Σ

Second reduction:

- Encode k-ary functions as $(k+1)$-ary relations
- Align the relation arities to be constantly $n+1$
- Merge relations into a single $(n+2)$-ary relation indexed by constants
- Interpret constants with fresh variables

Caveat: intermediate reductions may rely on discrete models...

Discrete Models

$\mathsf{FSAT}'(\Sigma)\,\varphi$ if $\mathsf{FSAT}(\Sigma)\,\varphi$ on a discrete model

Can every finite model $\mathcal{M}$ be transformed to a discrete finite model $\mathcal{M}'$?

Idea: first-order indistinguishability $x \doteq y \;:=\; \forall \varphi\rho.\, \mathcal{M} \vDash_{x\cdot\rho} \varphi \leftrightarrow \mathcal{M} \vDash_{y\cdot\rho} \varphi$

Lemma

The relation $x \doteq y$ is a decidable congruence for the symbols in Σ.

Fact

$\mathsf{FSAT}'(\Sigma)\,\varphi$ *iff* $\mathsf{FSAT}(\Sigma)\,\varphi$, *hence in particular* $\mathsf{FSAT}'(\Sigma)\,\varphi \preccurlyeq \mathsf{FSAT}(\Sigma)\,\varphi$.

Proof.

If $\mathcal{M} \vDash_\rho \varphi$ pick $\mathcal{M}'$ to be the quotient of $\mathcal{M}$ under $x \doteq y$. □

Compressing Relations: $\mathsf{FSAT}(\mathbb{O}; P^n) \preccurlyeq \mathsf{FSAT}(\mathbb{O}; \in^2)$

Intuition: encode $P\,x_1 \dots x_n$ as $(x_1, \dots, x_n) \in p$ for a set p representing P

So let's play set theory! For a set d representing the domain we define $\varphi'_\in$:

$$(P\,x_1 \dots x_n)'_\in := \text{``}(x_1, \dots, x_n) \in p\text{''} \qquad (\dot\forall z.\,\varphi)'_\in := \dot\forall z.\, z \in d \mathrel{\dot\rightarrow} (\varphi)'_\in$$

$$(\varphi \mathbin{\dot\Box} \psi)'_\in := (\varphi)'_\in \mathbin{\dot\Box} (\psi)'_\in \qquad (\dot\exists z.\,\varphi)'_\in := \dot\exists z.\, z \in d \mathbin{\dot\wedge} (\varphi)'_\in$$

Then $\varphi_\in$ is $\varphi'_\in$ plus asserting $\in$ to be extensional and d to be non-empty.

Fact

$\mathsf{FSAT}(\mathbb{O}; P^n)\,\varphi$ *iff* $\mathsf{FSAT}(\mathbb{O}; \in^2)\,\varphi_\in$, *hence* $\mathsf{FSAT}(\mathbb{O}; P^n) \preccurlyeq \mathsf{FSAT}(\mathbb{O}; \in^2)$.

Proof.

The hard direction is to construct a model of $\varphi_\in$ given a model $\mathcal{M}$ of φ. We employ a segment of the model of hereditarily finite sets by Smolka and Stark (2016) large enough to accommodate $\mathcal{M}$. □

Full Signature Classification

Composing all signature transformations verified we obtain:

Theorem

If Σ contains either an at least binary relation or a unary relation together with an at least binary function, then PCP *reduces to* $\mathsf{FSAT}(\Sigma)$.

On the other hand, FSAT for monadic signatures remains decidable:

Theorem

If Σ is discrete and has all arities bounded by 1 *or if all relation symbols have arity* 0*, then* $\mathsf{FSAT}(\Sigma)$ *is decidable.*

In any case, since one can enumerate all finite models up to extensionality:

Fact

If Σ is discrete and enumerable, then $\mathsf{FSAT}(\Sigma)$ *is enumerable.*

Example 3: First-Order Axiom Systems*

*Work in progress with Marc Hermes.

General Idea

Is a formula φ entailed by an axiomatisation A?

Strategy if A is strong enough to capture computation:

- Encode Turing machine M as formula φ_M
- Verify that M halts iff $A \vDash \varphi_M$
- Verify that M halts iff $A \vdash \varphi_M$ ($\rightarrow$ direction by hand)
- Instead of TM use problems suitable to encode in A

Connections to consistency and incompleteness:

- Reducing a non-trivial problem P to $A \vdash \varphi$ shows A consistent
- Undecidability implies incompleteness for enumerable axiomatisations

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

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

Proof.

If L has solution η instantiate the existential quantifiers with numerals $\overline{\eta_1}, \ldots, \overline{\eta_n}$. Then the axioms of PA entail the constraints.
If $\mathsf{PA} \vDash \exists^n c_1 \wedge \cdots \wedge c_k$ use the standard model $\mathbb{N}$ to extract solution η. □

Fact

$L = [c_1, \ldots, c_k]$ *with maximal index* x_n *is solvable iff* $\mathsf{PA} \vdash \exists^n c_1 \wedge \cdots \wedge c_k$.

Interlude: Models of ZF

Sets-as-trees interpretation (Aczel (1978)):

- Type $\mathcal{T}$ of well-founded trees with constructor $\tau : \forall X. (X \to \mathcal{T}) \to \mathcal{T}$
- Equality of trees s, t given by isomorphism $s \approx t$
- Membership defined by $s \in \tau\, X\, f := \exists x.\, s \approx f\, x$
- Set operations implemented by tree operations:
  - $\emptyset := \tau \perp \mathsf{elim}_\perp$
  - $\{s, t\} := \tau\, \mathbb{B}\, (\lambda b.\, \mathsf{if}\ b\ \mathsf{then}\ s\ \mathsf{else}\ t)$
  - $\omega := \tau\, \mathbb{N}\, (\lambda n.\, \overline{n})$ where $\overline{0} := \emptyset$ and $\overline{\mathsf{S}\, n} := \overline{n} \cup \{\overline{n}\}$
  - ...

Axioms needed in Coq:

- EM to really interpret ZF instead of IZF
- Replacement needs a type-theoretical choice axiom (Werner (1997))
- Strong quotient axiom for $(\mathcal{T}, \approx)$ suffices (Kirst and Smolka (2019))
- This yields a well-behaved model $\mathcal{S}$: quotiented, standard numbers

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: $S +\!\!+\, B := \bigcup_{s/t \in S} \{(\overline{s}x, \overline{t}y) \mid (x, y) \in B\}$
- $f \triangleright n := (\emptyset, \overline{S}) \in f \land \forall (k, B) \in f.\, k \in n \to (k + 1, S +\!\!+\, B) \in f$

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

Theorem

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

Proof.

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

Conclusion

Ongoing and Future Work

What I am involved with:

- Finish undecidability proofs for PA and ZF
- Extend and improve FOL mechanisation
  - Löwenheim-Skolem theorems, relative consistency proofs, etc.
  - Automated definability proofs, proof mode for ND derivations, etc.
  - Merge into a uniform core mechanisation

What other contributors are working on:

- Undecidability of semi-unification
- Undecidability of typability and type checking in System F
- Undecidability of IMSELL
- Verified compiler from cbv. lambda calculus to Turing machines
- Theoretical basis (e.g. consistency of computability axioms with EM)

Take-Home Messages

- Synthetic approach eases mechanisation of undecidability proofs
- Reductions (not only) to FOL benefit from using PCP
- Core reduction typically easy, remaining transformations intricate
- Constructive mechanisation of FOL rewarding but challenging
- If you work on undecidability proofs in Coq:
  Our library could help and is open for contributions

Thank You!

Bibliography

Aczel, P. (1978). The type theoretic interpretation of constructive set theory. In *Studies in Logic and the Foundations of Mathematics*, volume 96, pages 55–66. Elsevier.

Bauer, A. (2006). First steps in synthetic computability theory. *Electronic Notes in Theoretical Computer Science*, 155:5 – 31. Proceedings of the 21st Annual Conference on Mathematical Foundations of Programming Semantics (MFPS XXI).

Börger, E., Grädel, E., and Gurevich, Y. (1997). *The Classical Decision Problem*. Perspectives in Mathematical Logic. Springer-Verlag Berlin Heidelberg.

Forster, Y., Kirst, D., and Smolka, G. (2019). On synthetic undecidability in coq, with an application to the entscheidungsproblem. In *Proceedings of the 8th ACM SIGPLAN International Conference on Certified Programs and Proofs*.

Forster, Y., Kirst, D., and Wehr, D. (2020a). Completeness Theorems for First-Order Logic Analysed in Constructive Type Theory. In *Symposium on Logical Foundations Of Computer Science, 2020, Deerfield Beach, Florida, U.S.A.*

Forster, Y., Larchey-Wendling, D., Dudenhefner, A., Heiter, E., Kirst, D., Kunze, F., Smolka, G., Spies, S., Wehr, D., and Wuttke, M. (2020b). A Coq Library of Undecidable Problems. In *CoqPL 2020*, New Orleans, LA, United States.

Kirst, D. and Larchey-Wendling, D. (2020). Trakhtenbrot's theorem in coq, a constructive approach to finite model theory. *arXiv preprint arXiv:2004.07390*.

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.

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.

Libkin, L. (2010). *Elements of Finite Model Theory*. Springer Publishing Company, Incorporated, 1st edition.

Manna, Z. (2003). *Mathematical theory of computation*. Dover Publications, Inc.

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

Smolka, G. and Stark, K. (2016). Hereditarily Finite Sets in Constructive Type Theory. In *Interactive Theorem Proving - 7th International Conference, ITP 2016, Nancy, France, August 22-27, 2016*, volume 9807 of *LNCS*, pages 374–390. Springer.

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

Coq Mechanisation*

- Includes all results presented in the paper (PDF is hyperlinked!)
- Roughly 10k loc with additional 3k loc of utility libraries
  - More than 4k loc for $\mathsf{FSAT}(\mathbb{0}; P^n) \preccurlyeq \mathsf{FSAT}(\mathbb{0}; \in^2)$
  - Less than 500 loc for $\mathsf{PCP} \preccurlyeq \mathsf{FSATEQ}(\Sigma_{\mathsf{BPCP}}; \equiv)$
- FOL engineering similar to previous devs (cf. Forster et al. (2020a))
  - De Bruijn encoding of bound variables
  - Dependent syntax enforcing well-defined terms and formulas
- Axiom-free to ensure computability and interoperability
- Contributed to the Coq library of undecidability proofs†

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

* `https://www.ps.uni-saarland.de/extras/fol-trakh/`

First-Order Indistinguishability

We define operators $\mathrm{F}_{\mathcal{F}}, \mathrm{F}_{\mathcal{P}} : (D \to D \to \mathbb{P}) \to (D \to D \to \mathbb{P})$ by :

$$\mathrm{F}_{\mathcal{F}}(\mathcal{R})\, x\, y := \forall f.\, f \in l_{\mathcal{F}} \to \forall (\vec{v} : D^{|f|})\, (i : \mathbb{F}_{|f|}).\, \mathcal{R}\left(f^{\mathcal{M}}\, \vec{v}[x/i]\right)\left(f^{\mathcal{M}}\, \vec{v}[y/i]\right)$$
$$\mathrm{F}_{\mathcal{P}}(\mathcal{R})\, x\, y := \forall P.\, P \in l_{\mathcal{P}} \to \forall (\vec{v} : D^{|P|})\, (i : \mathbb{F}_{|P|}).\, P^{\mathcal{M}}\, \vec{v}[x/i] \leftrightarrow P^{\mathcal{M}}\, \vec{v}[y/i]$$

We then consider $\mathrm{F}(\mathcal{R}) := \mathrm{F}_{\mathcal{F}}(\mathcal{R}) \cap \mathrm{F}_{\mathcal{P}}(\mathcal{R})$ and show:

Theorem

First-order indistinguishability $\doteq$ up to $l_{\mathcal{F}}/l_{\mathcal{P}}$ is extensionally equivalent to $\equiv_{\mathrm{F}}$ (Kleene's greatest fixpoint of F), i.e. for any $x, y : D$ we have

$$x \doteq y \;\leftrightarrow\; x \equiv_{\mathrm{F}} y \quad \textit{where} \quad x \equiv_{\mathrm{F}} y \;:=\; \forall n : \mathbb{N}.\, \mathrm{F}^n(\lambda uv.\top)\, x\, y.$$

Moreover, the relation $x \equiv_{\mathrm{F}} y$ is decidable and hence so is $x \doteq y$.

Hereditarily Finite Sets

Theorem

Given a decidable n-ary relation $R : X^n \to \mathbb{P}$ over a finite, discrete and inhabited type X, one can compute a finite and discrete type Y equipped with a decidable relation $\in\, : Y \to Y \to \mathbb{P}$, two distinguished elements $d, r : Y$ and a pair of maps $i : X \to Y$ and $s : Y \to X$ s.t.

1. *$\in$ is extensional;*
2. *extensionally equal elements of Y are equal;*
3. *all n-tuples of members of d exist in Y;*
4. *$\forall x : X.\, i\, x \in d$;*
5. *$\forall y : Y.\, y \in d \to \exists x.\, y = i\, x$;*
6. *$\forall x : X.\, s(i\, x) = x$;*
7. *$R\, \vec{v}$ iff $i(\vec{v})$ is a n-tuple member of r, for any $\vec{v} : X^n$.*

Proof.

The type Y is built from the type of hereditarily finite sets. The idea is first to construct d as a transitive set of which the elements are in bijection i/s with the type X, hence d is the cardinal of X in the set-theoretic meaning. Then the iterated powersets $\mathcal{P}(d), \mathcal{P}^2(d), \ldots, \mathcal{P}^k(d)$ are all transitive as well and contain d both as a member and as a subset. Considering $\mathcal{P}^{2n}(d)$ which contains all the n-tuples built from the members of d, we define r as the set of n-tuples collecting the encoding $i(\vec{v})$ of vectors $\vec{v} : X^n$ such that $R\,\vec{v}$. We show $r \in p$ for p defined as $p := \mathcal{P}^{2n+1}(d)$. Then we define $Y := \{z \mid z \in p\}$ and restrict membership $\in$ to Y. □

Decidability Results

Lemmas used for decidability of monadic FOL and enumerability of FSAT:

Lemma

Given a discrete signature Σ and a discrete and finite type D, one can decide whether or not a formula over Σ has a (finite) model over D.

Lemma

A formula over a signature Σ has a finite and discrete model if and only if it has a (finite) model over $\mathbb{F}_n$ for some $n : \mathbb{N}$.