State of my Bachelor Thesis

The Undecidability of First-Order Logic over Small Signatures

Johannes Hostert

June 8, 2021

Contents

1 Introduction

It is known that various interesting properties of First-Order Logic (FOL) are undecidable. These include **Validity**, **Provability** and **Satisfiability**, which denote the property that a formula is satisfied in all models, is provable using a deduction system for FOL, or that there exists at least one model in which a formula is true. For a classical context, these problems have been shown undecidable by Church[2] and Turing[12]. In an intutionistic context like the one will be working in, they are still undecidable, but the proof is a bit more involved. The details can be found in Forster, et al [4]. We will use their development as a starting ground.

While these decision problems are undecidable in the general case, it might still be possible to decide them in special cases. For example, Löwenheim [9] showed that FOL is decidable as long as the used signature only has (at most) unary function and relation symbols. Contrarily, as soon as one introduces a function or relation with arity greater 1, the above problems become undecidable again [5]. This has already been mechanized in Coq in [6]

One usually shows this by giving several reductions, each of which "compresses" the signature, until eventually only a single binary relation remains. We, however, will show the undecidability of these problems for FOL with a signature containing exactly one binary relation using a single explicit reduction from a problem (i.e. a certain Diophantine constraints problem) not immediately related to FOL. We will formalize this in Coq, working within the Coq Library of Undecidability Proofs [3].

We will thus be working within the framework of synthetic computability theory [1]. This means that we will implement all our computable functions in Coq. Since all functions definable in Coq's type theory are computable, we do not need to show this explicitly. However, we cannot show that a certain function is undecidable this way. Thus we just assume that i.e. the halting problem is undecidable, and then formalize reductions from this problem.

2 Hilbert's 10th problem

In the following sections, we will be working with a discrete, countably infinite collection of variables $x, y, z, w, ... : \mathcal{V}$.

Hilbert's 10th problem famously asked for a decision procedure for whether a set of Diophantine equations has a solution. It later was established that this is undecidable [11]. A Diophantine equation is an equation over $\mathcal{V}$, where the variables are interpreted as integers and valid operations are $+, *$.

There are many variations of this problem, many of which are undecidable, too. One of these undecidable variations is the following:

The problem *UDC* is defined over a finite collection of triplets of $\mathcal{V}$:

$$\begin{aligned}&[(x_1, y_1, z_1), \dots, (x_n, y_n, z_n)] \in \mathit{UDC}\\ :\Leftrightarrow &\exists \varphi : \mathcal{V} \to \mathbb{N}, \forall 1 \leq i \leq n, 1 + \varphi x + (\varphi y)^2 = \varphi z\end{aligned}$$

The collection of tuples (x_i, y_i, z_i) can be understood as a list of Diophantine equations. Undecidability of this problem is already formalized in the library [8].

3 Uniform Diophantine Pair Constraints

We will first give another problem equivalent to *UDC*, which we will call **Uniform Diophantine Pair Constrains** (*UDPC*). A system of pair constraints is a list of $\mathcal{V}$ pair pairs $((a_i, b_i), (c_i, d_i)) : (\mathcal{V} * \mathcal{V}) * (\mathcal{V} * \mathcal{V})$. We first define the relation $(a, b)\#(c, d) :\Leftrightarrow 1 + a + b = c \wedge b \cdot (1 + b) = 2d$. The

decision problem *UDPC* is then defined as

$$\begin{aligned}&\{((a_1,b_1),(c_1,d_1)),\dots,((a_n,b_n),(c_n,d_n))\}\in \mathit{UDPC}\\ :\Leftrightarrow&\exists\varphi:\mathcal{V}\to\mathbb{N},\forall 1\le i\le n,(\varphi a_i,\varphi b_i)\#(\varphi c_i,\varphi d_i)\end{aligned}$$

This again can be understood as a collection of Diophantine equations.

We can show that this problem is undecidable by reducing from *UDC*. The reduction function can be given as

$$f(H) := \mathit{concat}\,[\,[((a,a),(b,t_1)),((c,y),(b,a)),((c,x),(z,t_2))]\,|\,(x,y,z)\in H\,]$$

where a,b,c,t_1,t_2 are fresh variables, and $\mathit{concat}:\mathscr{L}(\mathscr{L}X)\to\mathscr{L}X$. Verifying this in Coq is rather straightforward, it can be found in `theories/DiophantineConstraints/H10UC_to_UDPC.v`.

We remark two interesting properties of the relation #:

1. If $(a,0)\#(b,0)$, then $b = S\,a = a+1$. As a consequence, $\forall a:\exists b:(a,0)\#(b,0)$.
2. If $(a,b)\#(c,d)$, then $(a,S\,b)\#(S\,c,b+d+1)$. This can be reformulated into the following statement: $\forall abcdeb'c',(a,b)\#(c,d)\land(d,b)\#(e,d)\land(b,0)\#(b',0)\land(c,0)\#(c',0)\to(a,b')\#(c',e)$.

Both of these statements are formulated without reference to functions operating on integers, they only use variables and our relation #. This will become important in the next section.

4 Our First-Order instance and its models

As mentioned, we will show that validity, provability and satisfiability are undecidable for signatures with binary relations. We do this by constructing several many-one reductions from our seed problem *UDPC* towards a term in FOL over a signature $(\emptyset,\#)$ - that is, our siganture only contains the binary predicate #, and no functions.

Here, we overload the symbol #, which represents both as the relation on $\mathbb{N}^2$, and as a syntactic symbol for terms in the syntax of FOL over the aforementioned signature. To resolve ambiguities, we sometimes denote the latter as $\#_2$.

We will now gradually build up an FOL axiomatization of our # relation. At each step, we will consider the developed machinery in a "standard model". This standard model has objects $\mathcal{M} := \mathbb{N}^2+\mathbb{N}$, and $a\#^{\mathcal{M}}b$, the interpretation of $\#_2$ in $\mathcal{M}$ is defined as follows (with $n,m,x,y:\mathbb{N}$):

$$\begin{aligned}n\#^{\mathcal{M}}m &:= n=m\\ n\#^{\mathcal{M}}(z,w) &:= n=z\\ (x,y)\#^{\mathcal{M}}m &:= y=m\\ (x,y)\#^{\mathcal{M}}(z,w) &:= (x,y)\#(z,w)\end{aligned}$$

Since $\#^{\mathcal{M}}$ is an extension of # to $\mathcal{M}$ which is compatible with the original definition of #, we may also omit the $^{\mathcal{M}}$.

We now start with the following definitions:

$$\begin{aligned}N\,k :=& k\#k\\ P'\,k :=& N\,k\to\forall xy,x\#y\end{aligned}$$

$N\,k$ means that k is a number, because $k\#k$ only holds in this case. $P'\,k$ denotes that k is a pair. Note that $\forall xy,x\#y$ is contradictory in $\mathcal{M}$.

We now define:

$$P\,p\,l\,r\,\phi := P'\,p \to N\,l \to N\,r \to l\#p \to p\#_2 r \to \phi$$
$$rel\,p\,q\,a\,b\,c\,d\,\phi := P\,p\,a\,b\,(P\,q\,c\,d\,(p\#q \to \phi))$$

P denotes that $p = (l, r)$ in our standard model, i.e. p is a pair with l as the left and r as the right component. *rel* denotes that $(a, b)\#(c, d)$, where p, q are the pairs witnessing this.

We can now write our two properties from above as FOL terms:

$$\begin{aligned}
\varphi_1 :=& \forall n, N\,n \to \neg\forall lsr, rel\,l\,r\,n\,0\,s\,0 \to \bot \\
\varphi_2 :=& \forall p_1 p_2 p_3 p_4 p_5 p_6 p_7 p_8 abcde b'c', \\
& rel\,p_1\,p_2\,a\,b\,c\,d\,(rel\,p_3\,p_4\,d\,b\,e\,d\,(rel\,p_5\,p_6\,b\,0\,b'\,0\,(rel\,p_7\,p_8\,c\,0\,c'\,0\,(\psi)))), \\
& \text{where } \psi := \neg\forall lr, rel\,l\,r\,a\,b'\,c'\,e \to \bot
\end{aligned}$$

Note that we have encoded the $\exists$-quantifier as $\neg\forall\neg$. This is because our reduction actually targets the universal-implicative fragment of FOL, where the only allowed logical connectives are $\to, \forall, \neg$. We are also able to eliminate the $\neg$ by employing a Friedman translation, where we simply define:

$$\begin{aligned}
\bot :=& c_1 \#_2 c_2 \\
\neg\phi :=& \phi \to \bot
\end{aligned}$$

c_1 and c_2 are two variables bound somewhere. The same holds for 0 in the other terms above. Importantly, 0 is not understood as a nullary function.

Armed with these definitions, we can now define the final reduction function F, which, given an instance h of *UDPC*, gives a FOL statement:

$$single\,a\,b\,c\,d := \neg\forall lr, rel\,l\,r\,a\,b\,c\,d \to \bot$$

$$F[((a_1, b_1), (c_1, d_1)), \dots ((a_n, b_n), (c_n, d_n))] := \forall 0 c_1 c_2, N\,0 \to \varphi_1 \to \varphi_2 \to \neg \bigforall_{i:\mathcal{V}(h)} x_i\,, \psi, \text{ where}$$
$$m := 1 + \max\{a_1, b_1, c_1, d_1, \dots, a_n, b_n, c_n, d_n\}$$
$$\psi := single\,x_{a_1}\,x_{b_1}\,x_{c_1}\,x_{d_1} \to \dots \to single\,x_{a_n}\,x_{b_n}\,x_{c_n}\,x_{d_n} \to \bot$$

single denotes that $(a, b)\#(c, d)$. The complete reduction function first assumes the points c_1, c_2, and 0, then the axioms $N\,0, \varphi_1$, and φ_2. Then, behind a negation, we generate $\forall$-quantifiers which bind a new variable x_i for each variable used in the input. We then add a chain of preconditions, one for each constraint from our constraint set. Once these preconditions are met, we get $\bot$, allowing us to show the introduced negation.

As mentioned, our formula uses the minimal signature discussed above. Apart from that, we only have variables, $\forall$ and $\to$, which places us in the universal-implicative positive segment of FOL. This is desirable because it is a very weak form of FOL, so our undecidability result is stronger.

5 Validity

Formally, we show that it is undecidable whether a formula in our first-order logic is valid in all Tarski models by reducing from *UDPC*, using the above reduction functions. Now, two things remain to show for an arbitrary instance h of *UDPC*.

- If $h \in UDPC$, then $F(h)$ is valid in all models of L.
- If $F(h)$ is valid in all models of L, then $h \in UDPC$.

The second fact is the easier one, since, if $F(h)$ is valid in all models, it is valid in the standard model. For the actual proof, we use a slightly modified model - we define $n\#_2 m := n = m \vee (n = 0 \wedge m = 1 \wedge h \in UDPC)$. We will use this later by specifying c_1 and c_2 such that $\bot^{\mathcal{M}} \Leftrightarrow h \in UDPC$.

We then obtain that $f(h)$ is valid in the standard model. We specialize $F(h)$ with $0 := 0, c_1 := 0, c_2 := 1$. We then show that $N\,0$, φ_1 and φ_2 are indeed valid in this model. Finally, we arrive at the implication chain, which gives us all the necessary preconditions to actually show that h actually is $\in UDPC$, since $single\,a\,b\,c\,d$ is equivalent to $(a, b)\#(c, d)$ in our standard model.

The other direction is harder. We need to show that $F(h)$ is valid in any model M (with domain $D \neq \emptyset$). We assume that $N\,0, Ax1$ and $Ax2$ are valid for certain points $0, c_1, c_2 \in D$. We also get a φ which consistently assigns values to all "variables" in h, since $h \in UDPC$. We then need to establish two main lemmas before we can show that $F(h)$ is valid:

Lemma 1 (Chain)**.** *For any $n \in \mathbb{N}$, we can construct a "chain" $f : \mathbb{N} \to D$, which contains the "representation" of the first n numbers in D.*

Formally, we have: $N\,(f0) \wedge \forall 0 \leq i \leq n, \exists lr, \mathrm{l}$ is the pair $(fi, 0) \wedge \mathrm{r}$ is the pair $(f(i+1), 0) \wedge l\#r$.

Proof by induction on n.
We use the axiom φ_1 for the induction step. The base is shown by $N\,0$. □

Lemma 2. *For all pairs $(a, b), (c, d)$, if $(a, b)\#(c, d)$,*
and if we have a chain up to $\max\{\varphi a, \varphi b, \varphi c, \varphi d\}$ represented by f, then single $(fa)\,(fb)\,(fc)\,(fd)$ holds.

Proof by induction on b, generalized over a, b, d.
We use axiom φ_2 and the following straightforward inversion laws:

- $\forall acd, (a, 0)\#(c, d) \to d = 0 \wedge c = a + 1$
- $\forall abcd, (a, b + 1)\#(c, d) \to \exists c'd', (a, b)\#(c', d') \wedge c = c' + 1 \wedge (d', b)\#(d, d')$

Given a, b, c, d, if $b = 0$, then $d = 0, c = a + 1$, and we can use the constructed chain f. For the inductive step, we need to use the induction hypothesis twice, once for $(a, b)\#(c, d)$, and again for $(d, b)\#(e, d)$. □

This then allows us to find a proof of $\bot$ by first constructing a chain f up to $1 + m'$, where $m' = \max_{1 \leq i \leq m}, \varphi\, i$, then specializing $(\forall x_i)^m$ - quantification chain using $f(\varphi 0), f(\varphi 1) \ldots$. Then, the preconditions of kind $single\,x_{(a_i)}\,x_{(b_i)}\,x_{(c_i)}\,x_{(d_i)}$ must be true since $x_{a_i} = f(\varphi(a_i))$ and so on, and we can apply Lemma 2.

This then concludes the proof. As a result, it is undecidable in general whether a formula is valid in all models of FOL with our signature

6 Provability

Instead of investigating whether a formula ψ is valid in all models of our theory L, we can also investigate whether ψ is provable in the abstract deduction system of first-order logic. The specific system we are using denotes intuitionistic provability in the universal-implicative fragment without negation, because this again yields the strongest undecidability result. We write $\vdash \psi$ if a formula is provable in that system. In order to show that provability is undecidable, we use the same approach we used with validity. We again need to show both directions:

- If $h \in UDPC$, then $F(h)$ is provable in the proof system.
- If $F(h)$ has a proof in the proof system, then $h \in UDPC$.

The second direction is again far easier: Since the proof system is sound, we can assume that $F(h)$ is valid in the standard model. We've already shown that this implies $h \in UDPC$.

The concrete deduction system used is sound, which means that any formula proved with this system is valid in all models. This means that we can simply show that $h \in UDPC$ if $F(h)$ is provable, by reusing the earlier proof, by applying the soundness result.

Furthermore, it is quasi-complete, which means that for any formula ψ valid in all Tarski models, we can show that its negation (assuming we allow falsity in our FOL) is refutable. This means that, without additional axioms like Markov's principle or excluded middle, we can not simply conclude that our formula is provable if it is valid in all (Tarski) models. Furthermore, if we show that our formula is provable, we then also could conclude that our formula is valid in all Kripke models, whereas previously we only talked about Tarski models. Thus, showing that our formula has an abstract proof allows us to easily show other problems undecidable, too.

The general structure of constructing the abstract proof is similar to the earlier proof of "if $h \in UDPC$, then $F(h)$ is valid". We use similar lemmas, which can be shown with similar inductions, and so on.

However, we are now significantly constrained by our deduction system. Instead of having our chain represented by an $f : \mathbb{N} \to D$, along with the statement that "$\forall 1 \leq i \leq n, f(i)$ is a valid part of the chain", we are now working with an explicit list of hypotheses, which are in the syntax of first-order logic, which means we can not use functions explicitly in our formulas. This means that we must change the way we represent our chain - instead of a function along with a proof that certain pairs exist, we now give an explicit list containing the variables making up our chain. In our deduction system, we then have a list of hypotheses generated from our chain data which basically state "the variables used in the chain form a valid chain". While there are many different ways to represent the chain data, we use an index-inductive type, which has the upper bound n up to which the chain is valid as part of the type.

Armed with this representation, we can then state and prove the counterparts of Lemma 1 and 2 from the validity section:

Lemma 3 (Lemma 1 (Chain) for provability)**.** *For any n, $A \vdash \psi$ can be shown by showing that for any chain c up to heigth n, $chain\_exists\, c + +A \vdash \psi$.* Here, *chain_exists* generates the hypotheses stating that c is a valid chain, as discussed before.

Lemma 4 (Lemma 2 for provability)**.** *For all $abcd \in \mathbb{N}$, if cc is a chain containing a, b, c and d, and if $(a, b)\#(c, d)$, then $chain\_exists\, c + +A \vdash \bot$ can be shown by showing $chain\_exists\, c + +A \vdash single\, (fa)\, (fb)\, (fc)\, (fd)$*

These lemmata can then be proven similarly to the corresponding lemmata from the last section.

One can thus conclude that it is also undecidable whether a certain formula ψ of FOL over our signature can be proven in the corresponding deduction system.

7 Corollaries

By showing provability undecidiable, we can already subsume half of the proof we did before, because we can now show that $F(h)$ is valid in all models of L by showing that $F(h)$ is provable, and applying the soundness theorem.

Even further, we obtain a similar undecidability result for Kripke validity, i.e. it is undecidable whether a formula of our logic L is valid in all Kripke models. The reduction here is trivial, since one direction can be shown using the provability result, while the other direction can be shown by using the fact that if a formula is valid in all Kripke models, it is valid in all Tarski models, so one can use part of the proof from the validity section.

One can furthermore show that satisfiability is undecidable: A formula ψ of our logic L is satisfied if there exists a model M in which ψ holds. This can be shown by reducing from $\overline{UDPC}$, using the reduction function $F'(h) = \neg F(h)$ (where $\neg$ is native to the syntax). Then, if $F'(h)$ is satisfied, this means that there is a model where $F(h)$ does not hold, thus $F(h)$ is not valid, thus $h \in \overline{UDPC}$, since otherwise $F(h)$ must be valid. Similarly, if $h \in \overline{UDPC}$, then our standard model satisfies $F'(h)$, since if $F(h)$ was true, $h \in UDPC$ would hold.

A similar argument is likely to work for Kripke satisfiability.

8 Remarks on the Coq formalization

A major difference between our presentation here and the formalisation in Coq is that in Coq, de Bruijn indices are used instead of variables. For the proofs regarding the undecidability of validity, this is not a large issue, since interpreting these de Bruijn-formulas in a model resolves the indices to the usual binder scheme.

However, for the provability undecidability proof, this presented a significant hardship. A lot of infrastructure is needed to properly work with theorems, since one has to make sure the indices are manipulated properly.

Furthermore, when formalizing Lemma 1 (the "chain") for the provability reduction, it became important how the chain is represented. While we eventually settled on the approach of having a data structure which contain the de Bruijn indices of the parts of the chain, along with a list of hypotheses stating that these indices actually form a chain, we originally tried using specific indices for specific parts of the chain (e.g. the representation of number k in the chain is exactly the variable $3 * k + 1$). This made some parts of the proof easier. However, we ultimately decided against it, because it required more elaborate housekeeping to make sure all the indices are proper, which made the proof harder to understand.

9 Further work

The Coq formalization currently only has the reductions formalized. The actual undecidability results still need to be written down. Also, the formalization needs to be split up, modularized and refactored. Furthermore, it might be worth investigating whether the provability proof can be shortened, even though this seems unlikely: Despite this proof undergoing significant reworkings, it has not gotten shorter. We tried several ideas, and while we ended up making the proof simpler, it did not become much shorter, with all approaches having roughly the same number of lines of Coq code. We do not expect further refactors to change this. While more refactors might make the proof easier to read or understand, we do not expect significant improvement here, either. However, a recently developed Proof Mode for FOL [10], which aims at making these proofs simpler, might help here.

Other related properties that can be shown undecidable are finite satisfiability, which is similar to satisfiability, except that the model must be finite.

Also, it might be worth investigating whether a similar approach can be used to show that the discussed problems (validity, provability, ...) are undecidable for FOL over signatures with a binary function symbol (and an unary relation symbol).

10 Overview of related literature

10.1 Church 1936; Turing 1936

The paper [2] and [12] are seminal papers in computability theory. They establish the undecidability of the halting problem, and employ reductions to show that the Entscheidungsproblem as posed by Hilbert is undecidable. Formally, they show that it is undecidable whether a formula in FOL is valid. Turing's proof employs a signature containing at least 5 binary relations, though the actual number depends on the implementation details of the specific Turing machine one reduces from. Church's reduction partially formalizes first-order Peano arithmetic.

10.2 Kalmár 1937

Kalmár's paper [5] is part of "reduction theory", which tried to find "simple" FOL terms for arbitrary FOL terms. Here, "simple" can refer to the number or arity of the used relation symbols, or of the amount and alternation sequence of the used quantifiers. Kalḿar shows that for each FOL term $\mathfrak{U}$ over an arbitrary signature, there is an equivalent FOL $\mathfrak{B}$ term with an certain quantifier prefix and a single binary relation. His reduction performs syntax compression by finding a new binary relation, which can express the relation symbols in $\mathfrak{U}$. This compression step expands the domain of discourse. Interestingly, his reduction claims to also preserve finite satisfiability. The reduction heavily relies on previous work by Kalmar and others, which had already shown that for each formula there is an equivalent one with three binary relations.

10.3 Forster et al, 2019

The paper by Forster et al [4] mechanizes the result of Church [2] and Turing [12] in Coq. They work within the context of an intiuitionistic (meta)-theory. Their reduction does not consider the halting problem as the source problem, but instead uses Post's correspondence problem, which had already been mechanized in Coq [3]. Their reduction is already within the universal-implicative fragment, however, they use a signature consisting of one unary and two binary functions, as well as one nullary and two binary relations. They also cover classical provability by performing a Gödel-Gentzen-Friedman translation to construct an intuitionistic formula valid whenever the original formula was valid in classical logic.

10.4 Kirst and Hermes, 2021

The paper by Kirst and Hermes [6] mechanizes the undecidability of validity and (inutitionistic as well as classical) provability in several variants of Peano arithmetic and ZF set theory. Importantly, they obtain the result that ZF with a single binary relation is undecidable. Their result, however, is not within the universal-implicative fragment. They assume LEM to work with classical formulas.

10.5 Kirst and Larchey-Wendling, 2020

The paper by Kirst and Larchey-Wendling [7] mechanizes Trakhtenbrot's theorem in Coq. This theorem states that it is undecidable whether a first-order formula is finitely satisfiable as soon as there is a single binary relation. They show this by first showing that finite satisfiability over an arbitrary signature is undecidable, by reducing from PCP. Then they perform a sequence of signature reductions to eventually get a formula that has just one binary relation symbol. They also investigate differences between different formalisations of finite models in Coq, by considering models with and without an explicit equality operation.

References

[1] A. Bauer. First Steps in Synthetic Computability Theory. *Electronic Notes in Theoretical Computer Science*, 155:5–31, 2006. Proceedings of the 21st Annual Conference on Mathematical Foundations of Programming Semantics (MFPS XXI).

[2] A. Church. A note on the Entscheidungsproblem. *Journal of Symbolic Logic*, 1(1):4041, 1936.

[3] Y. F. et al. A Coq Library of Undecidable Problems. CoqPL 20, 2020.

[4] Y. Forster, D. Kirst, and G. Smolka. 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*, CPP 2019, page 3851, New York, NY, USA, 2019. Association for Computing Machinery.

[5] L. Kalmár. Zurückführung des Entscheidungsproblems auf den Fall von Formeln mit einer einzigen, binären, Funktionsvariablen. *Compositio Mathematica*, 4:137–144, 1937.

[6] D. Kirst and M. Hermes. Synthetic Undecidability and Incompleteness of First-Order Axiom Systems in Coq. In *Interactive Theorem Proving - 12th International Conference, ITP 2021, Rome, Italy*. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2021. To appear.

[7] D. Kirst and D. Larchey-Wendling. Trakhtenbrot's theorem in coq: A constructive approach to finite model theory. In *International Joint Conference on Automated Reasoning (IJCAR 2020), Paris, France*, Paris, France, 2020. Springer.

[8] D. Larchey-Wendling and Y. Forster. Hilbert's Tenth Problem in Coq. In H. Geuvers, editor, *4th International Conference on Formal Structures for Computation and Deduction (FSCD 2019)*, volume 131 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 27:1–27:20, Dagstuhl, Germany, 2019. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[9] L. Löwenheim. Über Möglichkeiten im Relativkalkül. *Mathematische Annalen*, 76:447–470, 1915.

[10] D. K. Mark Koch. A first order logic proof mode in Coq. `https://github.com/mark-koch/firstorder-proof-mode`. Accessed 2021-07-05.

[11] Y. V. Matiyasevich. Enumerable sets are Diophantine. *Doklady Akademii Nauk SSSR*, 191:279–282, 1970.

[12] A. M. Turing. On Computable Numbers, with an Application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, 2(42):230–265, 1936.