Perspectives for proof unwinding by programming languages techniques

Danko Ilik

PERSPECTIVES FOR PROOF UNWINDING BY PROGRAMMING LANGUAGES TECHNIQUES

DANKO ILIK

Abstract. In this chapter, we propose some future directions of work, potentially beneficial to Mathematics and its foundations, based on the recent import of methodology from the theory of programming languages into proof theory. This scientific essay, written for the audience of proof theorists as well as the working mathematician, is not a survey of the field, but rather a personal view of the author who hopes that it may inspire future and fellow researchers.

1. Introduction

> *We cannot hope to prove that every definition, every symbol, every abbreviation that we introduce is free from potential ambiguities, that it does not bring about the possibility of a contradiction that might not otherwise have been present.*
>
> N. Bourbaki [Bou49]

> *There is an error, I can confess now. Some 40 years after the paper was published, the logician Robert M. Solovay from the University of California sent me a communication pointing out the error. I thought: "How could it be?" I started to look at it and finally I realized* [...]
>
> John F. Nash Jr. [RS15]

> *Mathematics arises from all sorts of application or insights but in the end must always consist of proofs*, [but] *although a real proof is not simply a formalized document but a sequence of ideas and insights*, [a] *real proof is not something just probably correct.*
>
> Saunders Mac Lane [ML97]

What constitutes a *real proof* is a question at the origin of mathematical logic. In effect, a real proof is one that can be reduced to the use of only a few accepted 'ideal' principles such as the axioms for a set theory like ZFC. And yet certain ideal principles are far from self-evident as Euclid's axiomatic method would require them to be. Proof theory was conceived by Hilbert with the program to further "recognize the non-contradictory character of all the usual [ideal] mathematical methods without exceptions". Around 1960, these concerns were addressed for the theory of arithmetic and analysis in the so called *modified* Hilbert program using the early models of *computation*—proof theory was also pivotal for the development of computer science (Hilbert's Entscheidungsproblem).

Applying mathematical rigor to formal proofs as the object of study brought an answer to the question of what a real proof is: a formal proof can be given semantics in terms of Gödel's system T and Spector's bar recursion, thereby eliminating logic in favor of pure computation. However, these early models of computation that were used to provide the answer to the consistency question, although satisfying in terms of precision, are cumbersome to use in practice.

Firstly, it is far from clear why the old computational interpretations are the right ones, for it is often hard to distinguish them (bar recursion) from brute force search procedures. We would like to understand the computational answer to the main consistency questions in terms of modern and more finely grained computing abstractions, such as the ones developed over the course of the past four decades in the theory of programming languages—for research on (natural) models of computation surely did not end with the invention of the Turing machine and recursive function theory.

Secondly, the cumbersome machinery, although ingenious, makes it difficult to address the next level of research questions. Once that we have the answer to what a real proof *is*, we need to know what constitutes the *essential data* of a proof—curiously, this question of finding criteria of greatest simplicity for proofs was already listed as 24^{th} in Hilbert's famous list of open problems, but being premature was not included among the ones finally published [Thi03].

The title of this chapter refers to *proof unwinding*, the pioneering research program from the 1950's of Georg Kreisel [Luc96], who started to use the computational approach, not for foundational purposes, but to extract numerical content from actual mathematical arguments. We aim at the *working mathematician*, a term used by Bourbaki [Bou49] who meant by it a researcher with a pragmatic attitude toward foundations. The time is ripe for a leap forward, both in foundations and unwinding applications. The present chapter has as goal to propose bringing proof unwinding on a par with the latest computing abstractions from the theory of programming languages, with the ambition to turn such streamlined proof theoretic methods into a toolbox readily used by the working mathematician, rather than the rare specialist in proof theory as it has been the case up to now.

2. New Unwinding Toolbox

Conducted with the goal:

> *"To determine the constructive (recursive) content or the constructive equivalent of the non-constructive concepts and theorems used in mathematics"* [Kre58],

Kreisel's research program applied the proof theory of the day, namely Hilbert's ϵ-substitution method, Herbrand's theorem, and the no-counterexample interpretation, combined with then brand new theory of recursive functions, to extract new bounds and algorithms from prima facie ineffective proofs. But, even in the hands of masters, the early unwinding methodology was apparently difficult to apply, if one is to judge from the lapses of time in between applications: Littlewood's theorem by Kreisel in 1951 [Kre51], Artin's proof of Hilbert's 17^{th} problem by Kreisel first in 1957 and again in 1978 [Del96]), the Thue-Siegel-Roth theorem by Kreisel and Macintyre in 1982 and Luckhardt in 1989 [KM82, Luc89], Van der Waerden's theorem by Girard in 1987 [Gir87]. The unwinding methods are so complex that there are even doubts cast on some of the results by authorities in proof theory [Fef96].

However, there is a more recent application of unwinding to functional analysis in the *proof mining* program of Kohlenbach [Koh05]. This very successful unwinding program has at its methodological core the classic unwinding approach using Kolmogorov's double-negation translation (1929) and Gödel's functional 'Dialectica' interpretation (1941).

In parallel, in constructive mathematics, there have been equally significant results in the program of *constructive analysis* [BB85] and *constructive algebra* [MR88, LQ11], although these are primarily theory reconstruction programs and rely little on direct application of proof theoretic methods to unwind ineffective proofs.

But, both mining and research in constructive mathematics have not sought to reap the benefits of notable proof theoretic advances directly inspired by the theory of programming languages. This theory, a continuation of the work on the early models of computation, has arrived at highly abstract notions for structuring programs. We shall now describe the proof theoretic state-of-the-art for three such proof unwinding techniques. This new methodology will be referred to as the *unwinding toolbox*.

2.1. **Computational Side-Effects.** The first of these methods concerns *computational side-effect*. Namely, since the work of Griffin [Gri90], it has been known that the principle of proof by contradiction can be interpreted by a programming language mechanism (a computational side-effect) for *control operators*. Although, in absence of mathematical axioms additional to the reductio-ad-absurdum principle, control operators provide not much more than a very elegant way to obtain Herbrand's theorem (an important very early result on classical first-order logic from 1930), in presence of additional axioms like induction or choice, when Herbrand's theorem no longer holds, one begins to get new results. For instance, by the use of computational side-effects, in set theory, Krivine has managed to extend Cohen's forcing method from the usual sets of conditions to realizability algebras [Kri].

However, the promise that control operators can turn every proof by contradiction into an effective one is a mirage: there are classically provable formulas whose effective proof would allow to decide the Halting problem. Whether an ineffectively proved formula can be unwound, in general needs to be considered on a case-by-case basis. Nevertheless, there *are* whole *classes* of formulas which we know can be unwound upfront, like the class of Π^0_2-formulas. *Delimiting* control operators only to formulas in these classes allows to get a new constructive logic. This logic still respects the existence property, characteristic of intuitionistic logic that is at the bases of current constructive mathematics, but the obtained new constructive logic manages to prove intuitionistically non-provable principles.

For instance, Herbelin [Her10] showed that Markov's principle (MP),

$$\neg\neg\exists x A_0(x) \to \exists x A_0(x),$$

where $x \in \mathbb{N}$ and A_0 is quantifier-free, an axiom crucial for constructive proofs of completeness of first-order logic [Ili10], can be interpreted with the help of a computational side-effect known as *(delimited) exceptions*. The author further showed [Ili12] that the double negation shift principle (DNS),

$$\forall x \neg\neg A(x) \to \neg\neg\forall x A(x),$$

where $x \in \mathbb{N}$, a principle crucial for the interpretation of the classical axiom of choice—and that has only been interpreted before by the *generally*-recursive schema of bar recursion—can be interpreted computationally by a generalization of the exceptions effect to so called delimited continuations, or *delimited control operators*. The key observation from these results is that—when delimited—computational side-effects like control operators can be used to unwind ineffective proofs and at the same time not run into non-decidability problems. The newly obtained logics are intermediate logics, in between classical and intuitionistic logic, and take the best of both worlds.

These results are controversial from the point of view of the orthodox constructive mathematician who is used to intuitionistic logic as first codified by Heyting's analysis of Brouwer's work in intuitionistic mathematics. Namely, the only previous computational interpretation of MP were either trivial (as given by Gödel's functional interpretation) or proceeded by unbounded search. As for DNS, the computational interpretation was only given by the generally-recursive bar recursion schema, whose termination must be ensured by Brouwer's bar induction or continuity principle. As unbounded/general recursion can lead to an inconsistent formal system, intuitionists have been understandably wary of accepting these principle. By replacing the mentioned computational interpretations by computationally meaningful realizers, we not only propose to intuitionists to reconsider the constructivity of principles like MP and DNS, but we are re-establishing the link between modern proof theory and one of the offspring of Hilbert's proof theory, the theory of programming languages.

A further principle interpreted in this way, in a joint work of Nakata and the author [IN14], was the open induction principle,

$$\forall\alpha(\forall\beta < \alpha(\beta \in U) \to \alpha \in U) \to \forall\alpha(\alpha \in U),$$

where α, β range over Cantor space and U is open. This principle is the only known equivalent form of the axiom of choice that is stable under double-negation translation (even if we replace Cantor space by Baire space). The principle is also interesting for combinatorics, where it leads to a direct version of Nash-Williams' proof of Kruskal's tree theorem [Vel01], as well as in algebra where it is used to replace Zorn's lemma [Sch13].

We finally mention a last result [Ili14a], still in review, on the nature of the programming-language inspired proof rules. It concerns higher-type primitive recursion—Gödel's system T—versus general recursion—Spector's bar recursion. Namely, already in 1979, Schwichtenberg has shown that bar recursion of type 0 and 1 does not allow to define functions beyond system T [Sch79]. Since a previous analysis of Kreisel [Kre59] shows that these types are sufficient for all practical purposes (realizing Σ^0_2-theorems), it follows that we know for a long time that we should not need bar recursion for the computational interpretation of ideal proof principles. What we proposed is how to circumvent bar recursion and generate System T terms directly, using delimited control operators as an intermediary step. This also shows that the extensions of system T with computational side-effects are in fact conservative extensions. In order to establish this fact, we relied on *partial evaluation*, the second set of techniques of our unwinding toolbox that we explain in the following subsection.

2.2. **Partial Evaluation and Formalization.** The second programming languages method that we intend to employ for proof unwinding concerns formalization of proofs in proof assistant software and, more specifically, the use of formalized *(type-directed) partial evaluators*.

The topic of partial evaluators came up in our previous research on constructive versions of completeness theorems [Ili10]. These logical theorems establish the adequacy of a given formal system to encode actual mathematical proofs. As it turns out, and thanks to initial work on the link between normalization proofs and completeness of intuitionistic logic for Kripke models [CD97], the computational content of proofs of intuitionistic completeness can be expressed by type-directed partial evaluation algorithms [Dan96]. Having a rich theory of such algorithms in the theory of programming languages, allowed to cover cases of constructive completeness proofs that were beyond the previous state-of-the art in proof theory. More precisely, we now know how to partially evaluate (i.e. show constructively completeness for) not only classical logic [Ili12], intuitionistic logic with disjunction [Ili13a], but also simultaneous presence of delimited control operators and higher-type primitive recursion [Ili13b, Ili14a] (the second citation is still in review).

The development of these logical meta-theorems was conducted formally, in the Coq and Agda proof assistants. Since the formalized proofs are constructive, they can be used to *compute a proof transformation* for every actual formalized argument. What this allows is to perform unwinding of actual mathematical proofs more directly, by pushing the complexity of doing a manual double-negation transformation (like done in the classic unwinding approach and used, for instance, by Kohlenbach in his program of proof mining in analysis) into the realizability model, that is, into the reduction mechanism of the proof assistant used.

Proof assistants are most well known for their use in the full formalization of complex proofs, such as the four-color theorem [Gon08], the Kepler conjecture [Hal12], or the Feit-Thompson theorem [GAA$^+$13]. However, as far as proof unwinding is concerned, one can in general *avoid* needing a *fully* formalized version of an actual mathematical proof. It suffices to notice that lemmas that have a computationally irrelevant form, such as Π^0_1, can be simply assumed without proof. A more refined analysis of computational relevance of formulas can be found in [SW12] where the classes of so called definite and goal formulas are isolated. This allows to greatly decrease the burden of formalizing i.e. we are only dealing with *partial formalization* which nonetheless contains as much of algorithmic content as a full formalization.

The important lesson that we learned from partial evaluation is that proofs need not be interpreted uniformly, by 'oracles' such as bar recursion that work uniformly (for example the realization of DNS by bar recursion is agnostic of the concrete formula A in the instance of DNS). Rather, it is possible to specialize (i.e. partially evaluate) proofs, even if they are highly ineffective, and, when one in addition uses a proof assistant like Coq, the specialization of the (partially) formalized theorem can become automatic. This is one of the principal advantages of our toolbox over the old toolbox built on Herbrand's theorem, ϵ-substitution, double-negation- and A-translation, and functional interpretation: while unwinding, the mathematician can concentrate on the essential parts of a proof rather than get lost in manual proof transformations.

2.3. **Type Isomorphisms.** The final third method of our unwinding toolbox concerns *type isomorphisms*. Mathematically, this notion is the same as the one of

constructive cardinality of sets [MR88], saying not only that sets are of the same size, but moreover that they have indistinguishable structure. In programming languages theory, the notion allows to generalize the notion of type assigned to a program, which allows to test more easily if a programs conforms to a formal specification [Rit91].

The link that brings us to the study of type isomorphisms is Tarski's high-school algebra problem [BY04]. This basic question, asking whether the system of eleven arithmetic equation taught in high-school suffices to derive all the true equations between *exponential* polynomials, had taken some time to be answered in mathematical logic. It turned out that the high-school system is not complete, as shown by a counter-example of Wilkie in 1981 [Wil00], a true statement which is not derivable by only using the eleven equations. Gurevič further showed that the system cannot be completed by any additional finite set of axioms [Gur90].

Now, by the Curry-Howard correspondence, formulas of intuitionistic logic can be seen as types (conjunction correspond to products, disjunction to coproducts, and implication to exponentials) and proofs of formulas can be seen as computer programs of the corresponding type. Following the correspondence, one gets a notion of strong equivalence, or formula isomorphism, from isomorphism of types. A new correspondence is thus obtained: the language of formulas is the same as the language of exponential polynomials—and, moreover—formula isomorphism generalizes equality of exponential polynomials in the standard model of positive natural numbers, that is

$$A \cong B \text{ implies } \mathbb{N}^+ \vDash A = B.$$

The link that one establishes in this way allows to use the rich theory on exponential polynomials to obtain proof theoretic results. For instance, Fiore, Di Cosmo, and Balat, showed that the non-finite-axiomatizability result of Gurevič also hold for the theory of type isomorphism [FCB06]. Using results of Richardson, Martin, Levitz, Wilkie, Macintyre, Henson and Gurevič, the PI proved that although not finitely axiomatizable, type i.e. formula isomorphism is recursively axiomatizable and moreover decidable [Ili14b]. The value of this unexpected positive result is still somewhat limited because the existence of a *practical* decision algorithm for type isomorphism is open.

Nevertheless, even if the meta-theory of type isomorphism has remaining open questions to be resolved, applications to proof theory are well under way. Recently, the PI proposed a pseudo-normal form of types [Ili15], inspired by the decomposition of the exponential function in exponential fields [Har10], called the exp-log normal form, that allows to decompose the axioms of the notoriously non-local theory of $\beta\eta$-equality for the lambda calculus with coproduct type. This equality can be seen as the essence of *identity of proofs* for intuitionistic propositional logic with disjunction. An extension to the first-order case has also been proposed in a joint work with Brock-Nannestad [BNI16], where the normal form appears to produce the first arithmetical hierarchy for formulas of intuitionistic logic that copes with both quantifiers equally well; the only previously known hierarchy, the one of Burr [Bur00], covers well only the universal quantifier. This has been a long standing open problem for constructive logic, although for classical logic an arithmetical hierarchy exists since the 1930s.

3. Perspectives

Today, a paradigm change in proof unwinding is possible, thanks to the notions from contemporary programming languages theory comprising our New Unwinding Toolbox. These long-evolved techniques provide proof-theoretic simplifications of the order that makes them more accessible even to non-specialists in proof theory.

The overall goal of this undertaking would be to exploit the full potential of the novel toolbox and apply it, beyond logic itself, to proofs of landmark results in number theory, combinatorics, and homotopy theory. In parallel, it would be necessary to address the foundations of unwinding i.e. tackle long-standing open questions in the foundations of constructive mathematics such as identity of proofs and simplified computational interpretations of semi-intuitionistic principles. We have thus two sets of objectives, work on applications and work on foundations.

Objective I — Applications of Proof Unwinding. The first set of objectives concerns applications to areas that are important for the 'working mathematician', that is, analysis, number theory, and combinatorics, as well as an application to unwinding incompleteness theorems in logic. Objective I would be achieved by tackling three more specific objectives, called *perspectives*: Perspective 1: Unwinding in Analysis Revisited, Perspective 2: Unwinding in Number Theory and Combinatorics, and Perspective 3: Unwinding Incompleteness

Objective II — Foundations of Proof Unwinding. The second set of objectives concerns work on foundations of constructive mathematics that are both necessary to guarantee the soundness of applying unwinding and as an update to the current foundational theories. The two more specific objectives, or perspectives, to be tackled are: Perspective 4: Identity of Proofs and Homotopy Type Theory and Perspective 5: A Next Generation of Constructive Foundations.

The immediate effects of the project would be, on the one hand, to show that our new proof theoretic methods can be used by the working mathematician to extract numerical bounds and algorithms from prima facie ineffective proofs in analysis, number theory, combinatorics, homotopy theory, and logic, and, on the other hand, to update the current foundational theories of constructive mathematics with the powerful computing abstractions that computational side-effects, partial evaluators and type isomorphisms represent.

In the longer term, we can hope to see the streamlined proof unwinding methodology becoming an important toolbox across mathematics. We can also expect to see a synergy of the objectives. For instance, not only would unwinding efforts across mathematics become possible (Objective I), but, as the new constructive foundations (Objective II) get adopted in the community working on proof assistant systems, proof analysis and development would eventually be carried out even more efficiently with the help of a proof assistant.

The approach to fulfilling the two objectives would be through carrying out the five perspectives described in this section. We shall explain each one of the tasks in the context of its proper state-of-the-art, objectives, methodology, and feasibility.

3.1. **Perspective 1: Unwinding in Analysis Revisited.** Analysis is essentially the only area of mainstream mathematics to have benefited from direct application of proof unwinding techniques. In approximation theory, by using proof theory, Kohlenbach and his collaborators have managed to obtain explicit moduli of uniqueness, significantly better than previous ones, for best Chebyshev approximation, as well as to obtain a first effective rate of strong unicity in the case of best approximation for the L_1-norm [Koh93]. How this works is that first logical meta-theorems are established [Koh05], which are on one hand general enough to be applicable as analytic theorems, and on the other hand specific enough to enter in a class of statements, such as the Π^0_2-class of the arithmetical hierarchy, for which we know by proof theory that explicit functions or existence witnesses can be extracted. Moreover, such general logical meta-theorems are not only good for extracting numerical data from concrete proofs, but also for analyzing whether a known analytic theorem has optimal form. For instance, in the fixed point theory for functions of contractive type, one does not only get effective quantitative forms of theorems, but one can often also relax the compactness assumption for the metric space.

Why, then, when Kohlenbach's proof mining approach is already successful, do we propose a perspective on proof unwinding of analysis? There are two reasons. The first one is methodological: our form of unwinding has not been applied outside of logic, and proof mining provides the perfect test bed to make it grow up in the 'real world'. Second, even if we cannot pretend to analytic skills of the level of the ones present in mining, we do believe that the general logical meta-theorems can be unwound in a simpler way; this could lead to better extracted bounds even if we use the exact same analytic machinery as in mining.

To explain the difference and simplification mentioned, we briefly explain how the meta-theorems are established right now. The core is to show that in classical logic, and in presence of additional axioms for induction and choice, like the weak Kőnig's lemma, one can turn the $\forall\exists$ quantifier combination from $\forall x \exists y A_0(x, y)$, where A_0 is a quantifier-free formula, into an explicit recursive function f such that $\forall x A_0(x, f(x))$. One can further extend this to formulas beyond the strict class Π^0_2 and allow for instance any number of additional hypotheses of form Π^0_1. But, to obtain the recursive functions f, which, as explained before in the section Computational Side-Effects, needs an a priori generally recursive definition schema, one first has to transform by the double-negation translation all proofs of the original proof system (Peano arithmetic + axiom of choice) into proofs of a (semi-)intuitionistic system. This is a *non-local* transformation of proofs, and in particular the meaning of formulas can be changed by the transformation (hence the restriction to the Π^0_2-class of formulas). Once a (semi-)intuitionistic proof is obtained, one can use Gödel's functional interpretation to obtain a higher-type primitive recursive function, possibly also needing Spector's generally-recursive schema of bar recursion. Actually, more redefined versions of the Dialectica interpretation (monotone and bounded variants) and of bar recursion are used in practice.

Now, what our approach offers is first to push the technical complexity of the double negation translation into the realizability model based on computational side-effects (ex. control operators). Since these notions have a well-studied operational semantics, one can perform a more direct reduction of a proof to a program or a more direct reading off of witnesses (numeric bounds). With the additional help of a proof assistant like Coq, this can be further automatized.

In addition, thanks to the reasons already explained in section Computational Side-Effects, our unwinding method makes it likely that in fact a pure system T witnessing terms can be extracted from any concrete proof, circumventing bar-recursion-like schemata altogether.

A third, orthogonal, improvement to the extraction process will be offered by use of richer data-types for extracted programs and bounds. Traditionally, one only uses the 'negative' function and product types. Although these can encode 'positive' types (for instance, sum types $\rho+\sigma$ can be encoded by $(\mathbb{N} \to \rho)\times(\mathbb{N} \to \sigma)$), encoding leads to an increase of the *degree* of the type. Simpler and more natural realizers can thus be extracted from disjunction and other inductively defined positive predicates.

Feasibility for Perspective 1. We will need to cope with semi-intuitionistic principles that we have not treated before, notably the weak Kőnig's lemma and the independence of premise schema. For these, we plan to use Computational Side-Effects, like we have done previously for the open induction principle: the fan theorem, a positive version of the weak Kőnig lemma, is implied by open induction. At the level of realizers, it will be necessary to use Type Isomorphisms to handle extensionality.

The risk for handling the logical part (meta-theorems) is moderate, hence it is possible to propose this for a subject of a PhD thesis. As for obtaining better bounds that the ones already obtained in proof mining, the risk is higher; in fact, it would be a success even if we manage to obtain the same bounds, since this would mean that our toolbox is ready to be used in the following, Perspective 2.

3.2. **Perspective 2: Unwinding in Number Theory and Combinatorics.** In this task, we should bring in our New Unwinding Toolbox to bear on highly non-effective proofs from number theory and combinatorics. The concrete goals will be to unwind landmark proofs in these areas, but what we see as equally important is to arrive at a situation where a sufficiently interesting intersection of proof theory and the application domain area is recognized. This kind of objective is only possible through a combination of expertise, and for its carrying out, it would be appropriate to engage two post-doctoral researchers, one in each application domain.

In number theory, we would intend to unwind Thue-Siegel-Roth's theorem on Diophantine approximations. Saying that an algebraic irrational number has only finitely many exceptionally good rational number approximations, this Σ_2 statement has first been tackled upon by Kreisel and Macintyre [KM82] using technology for obtaining Herbrand terms. However the combinatorial explosion arising from use of Herbrand's theorem apparently did not allow to obtain useful bounds on the number of rational approximations, and only Luckhardt [Luc89] managed to limit the growth of Herbrand terms in order to obtain such a bound. This bound is essentially the same as the one obtained by Bombieri and van der Poorten [BVdP88].

In this case, even more advanced existing technology like Gödel's functional interpretation has not been applied. We suspect this is the case because, in order to apply it, one would first need to perform a double-negation translation of an actual proof of Thue-Siegel-Roth into a semi-intuitionistic theory—something possible to do in principle, but given the sophistication of the original proof, its translation would be an order of magnitude more complex. We propose thus to treat it directly using our approach with computational side-effects, i.e. without a preliminary double-negation translation followed by a functional interpretation.

Technically, our approach can be seen as a version of the so called modified realizability technique but where the language of realizers is enriched to contain delimited control operators.

In combinatorics, we would intend to unwind Szemerédi's theorem saying that every subset of the natural numbers with positive upper density contains arithmetic progressions of arbitrary length. Conjectured by Erdős and Turán in 1936, this statement was only proved by Szemerédi in 1975 by an ingenious and complex combinatorial argument. In 1977, Furstenberg provided a proof using ergodic theory. The interest in giving a better proof of this theorem is still ongoing, and applications include for instance the recent work of Green and Tao on arbitrary long arithmetic progressions in the prime numbers [GT08].

We first intend to address an important special case of the theorem, the van der Waerden theorem, saying that if we use a finite number of colors to color the natural numbers, then there is at least one color containing arbitrarily long arithmetic progressions. The current upper bound for van der Waerden's number $W(k, r)$, where r is the number of colors and k is the requested length of an arithmetic progression, was obtained via Szemerédi's theorem and is due to Gowers [Gow01]. What is intriguing in this subject is that the upper bounds appear to be heavy overestimates: for instance, the bound for $W(3, 3)$ is of the order of 10^{14616}, while the exact value is 27.

Girard has previously analyzed Furstenberg and Weiss's proof of van der Waerden's theorem using cut elimination [Gir87]. But, the bound that he arrived at was essentially the same upper bound obtained by Furstenberg and Weiss [Mac05]. We could attack the problem by using our modified realizability based on computational side-effects and attempt to partially evaluate the latest available proofs for Szemerédi's and van der Waerden's theorem—that would avoid the need for having a fully formalized proof on hand.

Feasibility for Perspective 2. Although the *statements* of the mentioned theorems in number theory and combinatorics are arithmetical, their *proofs* are not arithmetical. The risk for the objectives of this task is to cope with the highly non-effective nature of proofs, as well as their considerable complexity (see Szemerédi's diagram of lemmas from his proof in [Sze75]). After all, proofs of the corresponding theorems have brought Fields medals to both Roth and Szemerédi. The main proof theoretic question is which kind of ideal principles are at the core of arguments and can we provide a direct constructive justification for them. Sometimes, as in the case of Kruskal's theorem, another statement of Ramsey theory, the link to the open induction principle (analyzed previously in joint work with Nakata [IN14]) turns out to be direct [Vel01].

We intend to use proof assistant technology and partial formalization to cope with the complexity of proofs. Concerning mathematical risk, given a choice of motivated post-doctoral researchers to work on this topics, I would say that it is medium. Work on ergodic theory done in the previous Perspective 1: Unwinding in Analysis Revisited would serve as preparatory work and would help to further mitigate the risk. This task demands more resources than the other ones.

3.3. **Perspective 3: Unwinding Incompleteness Theorems.** A statement is said to be independent from a theory if it can neither be proved nor disproved from the axioms of the theory. The *incompleteness* phenomenon is the fact that for *any*

theory, under the assumption that it is consistent, there exist statements that are independent of the theory. One might wonder what is the nature of these statements, and whether they are relevant in practice. Indeed, the first such statement discovered by Gödel in 1931 has an 'artificial' flavor since it encodes the Epimenides' liar paradox. But, later, natural examples from Ramsey theory have been found, first by Paris and Harrington [PH77], and include important results like Kruskal's tree theorem. Finding concrete mathematical incompleteness statements is nowadays a fruitful field of research led by Friedman [Fri15].

However, what we find especially interesting is the *limit* at which a statement starts to become independent from a theory. This phenomenon, called *phase transition* by analogy to thermodynamics, happens when the *provability* of a theorem, taking a rational number as parameter, depends on the *value* of this rational parameter. For instance, a parametrized version of Kruskal's theorem can be provable in Peano arithmetic (PA) below a certain value of the parameter, and becomes independent above that value—this is in fact a real number, often found by use of analytic combinatorics, and provides a measure of the strength of the axiom system. Phase transitions are a research program led by Weiermann [Wei05].

In this task, we propose to develop a method for unwinding incompleteness theorems and phase transition phenomena based on programming language theory. The idea is that an incompleteness theorem, $\mathrm{PA} \nvdash \bot \to \mathrm{PA} \nvdash \mathrm{Con(PA)}$, saying that no consistent formal system (in this case, Peano arithmetic (PA)) can prove its own consistency, can be rephrased positively as $\mathrm{PA} \vdash \mathrm{Con(PA)} \to \mathrm{PA} \vdash \bot$. Translated in programming languages terms, $\mathrm{PA} \vdash \mathrm{Con(PA)}$ expresses the possibility of writing an interpreter for Gödel's system T inside system T itself—that is, a *self-interpreter*. Self-interpreters have not only been studied in programming languages theory, but they are a standard way to bootstrap a compiler for a programming language.

Nevertheless, self-interpreters are usually written for a Turing-complete languages like Scheme and ones without a strong typing discipline. If one adds a type system on top of Scheme one can retrieve system T in its λ-calculus formulation. There are recent intriguing results on typed self-interpreters. Brown and Palsberg have recently constructed the first *typed* self-interpreter [BP15]; their target was Girard's system U, and this is still 'acceptable', since system U is known to be inconsistent as a logical system. But, their latest result concerns Girard's system F_ω [BP], which is a higher-order logic and considered to be consistent.

In this task, we would first investigate whether it is possible to construct a self-interpreter for system T. For the purpose of the paper [Ili14a], we have already developed a formally verified interpreter for system T^+ inside Martin-Löf type theory. Since this type theory has a realizability model based on system T, one comes close to having a self-interpreter. We would also have to study the recent results of Brown and Palsberg, and attempt to retrieve their result for system F_ω in system T.

Feasibility for Perspective 3. The proposed methodology involves a frontal attack on consistency of PA. Although the risk is high, the fact that the prior works of Brown and Palsberg, and the PI, all involve formally checked proofs, gives us some confidence. If our effort succeeds, the gain one may have will be equally high as the risk. But, even if it turns out to be impossible to write a typed self-interpreter for T, we can aim to obtain solid results on interpreting Weiermann's phase transition, and hence characterizing the strength of a formal system, in terms of notions that

are equally natural from the point of view of computation as analytic combinatorics are.

3.4. **Perspective 4: Identity of Proofs and Homotopy Type Theory.** Formal proofs are combinatorial objects meant to encode a fully correct mathematical argument, going down to the smallest details, but that makes it difficult to spot the most essential parts of an arguments. Curiously, finding "criteria of simplicity, or proof of the greatest simplicity of certain proofs" was already part of Hilbert's program, who even planned to include it as the 24^{th} in his famous list of open problems [Thi03]. In particular, Hilbert asked for a procedure to decide when two given proof are essentially the same. This problem known as *identity of proofs* is still open [Dos03].

In this task, we would start by tackling the identity of proofs for constructive logic, before proceeding to a vast generalization of it, the computational interpretation of Voevodsky's univalence axiom in homotopy type theory [Coq14] in the case when the underlying definitional equality has been strengthened to decide identity of proofs i.e. to convertibility modulo isomorphism for dependent types.

For intuitionistic *propositional* logic, the difficulty of deciding identity of proofs comes from the simultaneous presence of disjunction and implication. Nevertheless, if we follow the analogy between formulas, types, and exponential polynomials, explained in section Type Isomorphisms, we can re-express the problem precisely as that of the effective decidability of the $\beta\eta$-equational theory for the lambda calculus with coproducts. We have recently proposed a first step in this direction by showing how to decompose the equational theory for terms, by the use of the exp-log normal form for types in order to enlarge the $\beta\eta$-congruence classes of terms [Ili15] (in review).

This exp-log normal form of types is extensible to the *first-order* case, when the quantifiers $\forall$ and $\exists$ are also present. Namely, recent work with Brock-Nannestad [BNI16] shows that it leads to an intuitionistic arithmetical hierarchy, a classification of formulas that was elusive for intuitionistic logic, even though it has existed for classical logic since the 1930's where it is at the basis of results like the completeness theorem.

A further question is whether we can make the technique work for dependent types, an extension of the first-order case. Martin-Löf Type Theory has dependent types which allow it to have special treatment of equality. Basic equality between elements a, a' of a type A is encoded by the identity type for A, $\mathrm{Id}_A(a, a')$. Identity of proofs in this context means extending the notion of definitional (computational) equality to cope with η-equality for coproducts (and other inductive types).

Pursuing generalization even further, we can talk about *identity between proofs of identity*, $\mathrm{Id}_{\mathrm{Id}_A}(p, p')$, that, in turn, endows ever type A with the structure of a groupoid. Iterating this construction, $\mathrm{Id}_{\mathrm{Id}_{\mathrm{Id}_{\cdots}\,\mathrm{Id}_A}}$, allows to show that every type A is in fact endowed with the structure of an ∞-groupoid [HS98]. Using Grothendieck's correspondence between ∞-groupoids and homotopy types has led Voevodsky to give a homotopy theoretic interpretation of type theory in his model based on simplicial sets [KLV12]. This model satisfies Voevodsky's *univalence axiom*, generalizing identity of proofs, and specializing to: equality at the level of propositions, bijection at the level of sets, categorical equivalence at the level of groupoids, etc. Adding this axiom on top of Martin-Löf's type theory produces homotopy type

theory, which is a logical system formalizing the *univalent foundations* of mathematics [Uni13].

What we propose to do is to build the convertibility of proof terms modulo type isomorphism into the definitional equality of Martin-Löf and homotopy type theory. An identity type then gets to cover equality between terms of a whole class of isomorphic types instead of only one type. We hope that in this way it will be possible to strengthen the notion of *transport of structures* and to show that important special cases of the univalence axioms satisfy a simple computational interpretation. The only existing computational interpretation of homotopy type theory appears in the effective version of the simplicial set model [BCH14] and works for the standard (restricted) notion of identity type.

Feasibility for Perspective 4. The univalence axiom is known to imply a form of full functional extensionality in type theory. Given that extensionality of functions in general is undecidable, the risk for extending the computational interpretation for the univalence axiom defined over the notion of identity types strengthened to work modulo isomorphism is high. Nevertheless, by strengthening the underlying definitional equality of the type theory, we hope to diminish the need for resorting to full functional extensionality and even address important special cases of univalence more simply than before.

As concerns the identity of proofs for the propositional and first-order case, based on our preliminary investigations of this area, we would say that the risk is moderate.

3.5. **Perspective 5: A Next Generation of Constructive Foundations.** This task would serve as an umbrella for more specific but important problems that need to be tackled in the foundations of constructive mathematics, as well as an umbrella collecting the foundational implications of the previous four tasks of this chapter.

For instance, we already know that axioms which are independent of intuitionistic logic like double negation shift can be safely added to intuitionistic systems, but we have to establish the outer limits of the potential given by Computational Side-Effects. We need to develop direct computational interpretations of principles arising from the work in constructive reverse mathematics, such as the equivalent forms of the open induction principle [Vel14], the extension of our work [IN14] to Baire space, and novel versions of Markov's principle [FIN15].

Another important topic will be to provide a direct constructive proof of Goodman's theorem. This theorem says that the axiom of choice presents a conservative extension of higher type Heyting arithmetic concerning arithmetical formulas; for the meta-theory of constructive mathematics, it plays the role that Hilbert's ϵ-elimination theorems play for the proof theory of classical logic. There has recently been renewed interest about this old result of Goodman by other researchers as well [Koh99, Coq13].

A third important topic will be to find practical decision algorithms for type isomorphism. As explain in the section Type Isomorphisms, although a decidability result holds for type isomorphisms [Ili14b], thanks to prior work of Richardson [Ric69] and Macintyre [Mac81], it is not clear at the moment whether a (practical) decision algorithm can be constructed. Arriving at such an algorithm would not only be useful for proof theory, but also for symbolic computation.

Finally, we would like to interact with the researchers working on proof assistant systems like Coq. The logical cores of proof assistants are lagging behind contemporary proof theory. For instance, program extraction from proofs in a state-of-the-art proof assistants such as Coq relies on the simplest possible realizability interpretation, the so called modified realizability interpretation of Kreisel. Integrating the techniques from the New Unwinding Toolbox would be beneficial for users of proof assistants because it would allow for easier formalization of many apparently ineffective proofs.

3.5.1. *Feasibility for Perspective 5.* The main challenge for this task is that, when we are interpreting semi-intuitionistic principles, we are working at the limit of computability: our realizability models for the classical axiom of choice refute the internal (formal) version of Church's thesis, but the external weak Church's rule still holds [Ili14a] (in review). It is thus hard to predict upfront how far the outer limits of constructive foundations can be extended. As concerns Goodman's theorem, we think the risk involved is not very high, since after all this result has been establish by non-direct methods. Finally, the risk on finding a practical algorithm deciding type isomorphism is hard to estimate; but even if we manage to find ones that only work for special cases, the benefits could spread also beyond proof theory.

References

[BB85] Errett Bishop and Douglas S. Bridges. *Constructive Analysis*, volume 279 of *Grundlehren der mathematischen Wissenschaften*. Springer-Verlag Berlin Heidelberg, 1985.

[BCH14] Marc Bezem, Thierry Coquand, and Simon Huber. A model of type theory in cubical sets. In *19th International Conference on Types for Proofs and Programs (TYPES 2013)*, volume 26, pages 107–128, 2014.

[BNI16] Taus Brock-Nannestad and Danko Ilik. An intuitionistic formula hierarchy based on high-school identities. *arXiv:1601.04876*, 2016. Submitted.

[Bou49] N. Bourbaki. Foundations of mathematics for the working mathematician. *The Journal of Symbolic Logic*, 14(1):1–8, 1949.

[BP] Matt Brown and Jens Palsberg. Breaking through the normalization barrier: A self-interpreter for F-omega. To appear in Proceedings of the 43rd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages.

[BP15] Matt Brown and Jens Palsberg. Self-representation in Girard's system U. In *Proceedings of the 42nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 471–484. ACM, 2015.

[Bur00] Wolfgang Burr. Fragments of Heyting arithmetic. *The Journal of Symbolic Logic*, 65(3):1223–1240, 2000.

[BVdP88] E Bombieri and AJ Van der Poorten. Some quantitative results related to roth's theorem. *Journal of the Australian Mathematical Society (Series A)*, 45(02):233–248, 1988.

[BY04] Stanley N. Burris and Karen A. Yeats. The saga of the high school identities. *Algebra Universalis*, 52:325–342, 2004.

[CD97] Thierry Coquand and Peter Dybjer. Intuitionistic model constructions and normalization proofs. *Mathematical Structures in Computer Science*, 7(1):75–94, 1997.

[Coq13] Thierry Coquand. About Goodman's theorem. *Annals of Pure and Applied Logic*, 164(4):437–442, 2013.

[Coq14] Thierry Coquand. Théorie des types dépendants et axiome d'univalence. *Séminaire BOURBAKI*, 66(1085), 2014.

[Dan96] Olivier Danvy. Type-directed partial evaluation. In *Proceedings of the Twenty-Third Annual ACM SIGPLAN SIGACT Symposium on Principles of Programming Languages (POPL'96)*, pages 242–257, 1996.

[Del96] Charles N. Delzell. Kreisel's unwinding of Artin's proof. In Piergiorgio Odifreddi, editor, *Kreiseliana. About and Around Georg Kreisel*, pages 113–246. A K Peters, 1996.

[Došo3] Kosta Došen. Identity of proofs based on normalization and generality. *Bulletin of Symbolic Logic*, 9(4):477–503, 2003.

[FCB06] Marcelo Fiore, Roberto Di Cosmo, and Vincent Balat. Remarks on isomorphisms in typed lambda calculi with empty and sum types. *Annals of Pure and Applied Logic*, 141:35–50, 2006.

[Fef96] Solomon Feferman. Kreisel's "Unwinding" Program. In Piergiorgio Odifreddi, editor, *Kreiseliana. About and Around Georg Kreisel*, pages 247–273. A K Peters, 1996.

[FIN15] Makoto Fujiwara, Hajime Ishihara, and Takako Nemoto. Some principles weaker than Markov's principle. *Archive for Mathematical Logic*, 54(7-8):861–870, 2015.

[Fri15] Harvey Friedman. *Boolean Relation Theory and Incompleteness*. Lecture Notes in Logic. ASL Publications, 2015.

[GAA+13] Georges Gonthier, Andrea Asperti, Jeremy Avigad, Yves Bertot, Cyril Cohen, François Garillot, Stéphane Le Roux, Assia Mahboubi, Russell O'Connor, Sidi Ould Biha, Ioana Pasca, Laurence Rideau, Alexey Solovyev, Enrico Tassi, and Laurent Théry. A Machine-Checked Proof of the Odd Order Theorem. In Sandrine Blazy, Christine Paulin, and David Pichardie, editors, *ITP 2013, 4th Conference on Interactive Theorem Proving*, volume 7998 of *LNCS*, pages 163–179, Rennes, France, July 2013. Springer.

[Gir87] Jean-Yves Girard. *Proof theory and logical complexity*, volume 1. Bibliopolis, Naples, 1987.

[Gon08] Georges Gonthier. Formal proof—the four-color theorem. *Notices of the AMS*, 55(11):1382–1393, December 2008.

[Gow01] W.T. Gowers. A new proof of Szemerédi's theorem. *Geometric & Functional Analysis GAFA*, 11(3):465–588, 2001.

[Gri90] Timothy G. Griffin. A formula-as-types notion of control. In *Conf. Record 17th Annual ACM Symp. on Principles of Programming Languages, POPL'90, San Francisco, CA, USA, 17-19 Jan 1990*, pages 47–58, 1990.

[GT08] Ben Green and Terence Tao. The primes contain arbitrarily long arithmetic progressions. *Annals of Mathematics*, pages 481–547, 2008.

[Gur90] R. H. Gurevič. Equational theory of positive numbers with exponentiation is not finitely axiomatizable. *Annals of Pure and Applied Logic*, 49:1–30, 1990.

[Hal12] Thomas Hales. *Dense sphere packings: A blueprint for formal proofs*, volume 400 of *London Mathematical Society Lecture Note Series*. Cambridge University Press, 2012.

[Har10] Godfrey Harold Hardy. *Orders of Infinity. The 'Infinitärcalcül' of Paul Du Bois-Reymond*. Cambridge Tracts in Mathematic and Mathematical Physics. Cambridge University Press, 1910.

[Her10] Hugo Herbelin. An intuitionistic logic that proves Markov's principle. In *Proceedings of the 25th Annual IEEE Symposium on Logic in Computer Science, LICS 2010, 11-14 July 2010, Edinburgh, United Kingdom*, pages 50–56. IEEE Computer Society, 2010.

[HS98] Martin Hofmann and Thomas Streicher. The groupoid interpretation of type theory. In *Twenty-five years of constructive type theory (Venice, 1995)*, volume 36 of *Oxford Logic Guides*, pages 83–111. Oxford Univ. Press, New York, 1998.

[Ili10] Danko Ilik. *Constructive Completeness Proofs and Delimited Control*. PhD thesis, École Polytechnique, Palaiseau, France, October 2010.

[Ili12] Danko Ilik. Delimited control operators prove double-negation shift. *Annals of Pure and Applied Logic*, 163(11):1549 – 1559, 2012.

[Ili13a] Danko Ilik. Continuation-passing style models complete for intuitionistic logic. *Annals of Pure and Applied Logic*, 164(6):651 – 662, 2013.

[Ili13b] Danko Ilik. Type directed partial evaluation for level-1 shift and reset. In Ugo de'Liguoro and Alexis Saurin, editors, Proceedings First Workshop on *Control Operators and their Semantics,* Eindhoven, The Netherlands, June 24-25, 2013 , volume 127 of *Electronic Proceedings in Theoretical Computer Science*, pages 86–100. Open Publishing Association, 2013.

[Ili14a] Danko Ilik. An interpretation of the Sigma-2 fragment of classical Analysis in System T. *arXiv:1301.5089*, 2014. Submitted.

[Ili14b] Danko Ilik. Axioms and decidability for type isomorphism in the presence of sums. In *Proceedings of the Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 53:1–53:7. ACM, 2014.

[Ili15] Danko Ilik. On the exp-log normal form of types. *arXiv:1502.04634*, 2015. Submitted.

[IN14] Danko Ilik and Keiko Nakata. A direct version of Veldman's proof of open induction on Cantor space via delimited control operators. *Leibniz International Proceedings in Informatics (LIPIcs)*, 26:188–201, 2014.

[KLV12] Chris Kapulkin, Peter LeFanu Lumsdaine, and Vladimir Voevodsky. The simplicial model of univalent foundations. *arXiv preprint arXiv:1211.2851*, 2012.

[KM82] Georg Kreisel and Angus MacIntyre. Constructive logic versus algebraization I. In A.S. Troelstra and D. van Dalen, editors, *The L.E.J. Brouwer Centenary Symposium*, pages 217–260. North-Holland Publishing Company, 1982.

[Koh93] Ulrich Kohlenbach. New effective moduli of uniqueness and uniform a priori estimates for constants of strong unicity by logical analysis of known proofs in best approximation theory. *Numerical Functional Analysis and Optimization*, 14(5-6):581–606, 1993.

[Koh99] Ulrich Kohlenbach. A note on Goodman's theorem. *Studia Logica*, 63(1):1–5, 1999.

[Koh05] Ulrich Kohlenbach. Some logical metatheorems with applications in functional analysis. *Transactions of the American Mathematical Society*, 357(1):89–128, 2005.

[Kre51] Georg Kreisel. On the interpretation of non-finitist proofs—Part I. *The Journal of Symbolic Logic*, 16(04):241–267, 1951.

[Kre58] Georg Kreisel. Mathematical significance of consistency proofs. *The Journal of Symbolic Logic*, 23(02):155–182, 1958.

[Kre59] Georg Kreisel. Interpretation of analysis by means of constructive functionals of finite types. In Arend Heyting, editor, *Constructivity in Mathematics, Proceedings of the colloqium held at Amsterdam, 1957*, Studies in Logic and The Foundations of Mathematics, pages 101–127. North-Holland Publishing Company Amsterdam, 1959.

[Kri] Jean-Louis Krivine. On the structure of classical realizability models of ZF. To appear.

[LQ11] Henri Lombardi and Claude Quitté. *Algèbre commutative – Méthodes constructives*. Calvage & Mounet, Paris, 2011.

[Luc89] Horst Luckhardt. Herbrand-Analysen zweier Beweise des Satzes von Roth: Polynomiale Anzahlschranken. *The Journal of Symbolic Logic*, 54(01):234–263, 1989.

[Luc96] Horst Luckhardt. Bounds Extracted by Kreisel From Ineffective Proofs. In Piergiorgio Odifreddi, editor, *Kreiseliana. About and Around Georg Kreisel*, pages 289–300. A K Peters, 1996.

[Mac81] Angus Macintyre. *Model Theory and Arithmetic*, volume 890 of *Lecture Notes in Mathematics*, chapter The laws of exponentiation, pages 185–197. Springer Berlin Heidelberg, 1981.

[Mac05] Angus Macintyre. The mathematical significance of proof theory. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 363(1835):2419–2435, 2005.

[ML97] Saunders Mac Lane. Despite physicists, proof is essential in mathematics. *Synthese*, 111(2):147–154, 1997.

[MR88] Ray Mines and Fred Richman. *A course in constructive algebra*. Springer, 1988.

[PH77] Jeff Paris and Leo Harrington. A mathematical incompleteness in Peano arithmetic. *Handbook of mathematical logic*, 90:1133–1142, 1977.

[Ric69] Daniel Richardson. Solution of the identity problem for integral exponential functions. *Zeitschrift für mathematische Logik und Grundlagen der Mathematik*, 15:333–340, 1969.

[Rit91] Mikael Rittri. Using types as search keys in function libraries. *Journal of Functional Programming*, 1:71–89, 1991.

[RS15] Martin Raussen and Christian Skau. Interview with Abel laureate John F. Nash Jr. *European Mathematical Society. Newsletter*, 97:26–31, September 2015.

[Sch79] Helmut Schwichtenberg. On bar recursion of types 0 and 1. *The Journal of Symbolic Logic*, 44(3), 1979.

[Sch13] Peter Schuster. Induction in algebra: A first case study. *Logical Methods in Computer Science*, 9(3):1–19, 2013.

[SW12] Helmut Schwichtenberg and Stanley S. Wainer. *Proofs and Computations*. Perspectives in Logic. Cambridge University Press, 2012.

[Sze75] Endre Szemerédi. On sets of integers containing no k elements in arithmetic progression. *Acta Arith*, 27(199-245):2, 1975.

[Thi03] Rüdinger Thiele. Hilbert's twenty-fourth problem. *American Mathematical Monthly*, 2003.

[Uni13] The Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations of Mathematics*. `http://homotopytypetheory.org/book`, Institute for Advanced Study, 2013.

[Vel01] Wim Veldman. An intuitionistic proof of Kruskal's theorem. *Archive for Mathematical Logic*, 43:215–264, 2001.

[Vel14] Wim Veldman. The principle of open induction on Cantor space and the approximate-fan theorem. *arXiv preprint 1408.2493*, 2014.

[Wei05] Andreas Weiermann. Analytic combinatorics, proof-theoretic ordinals, and phase transitions for independence results. *Annals of Pure and Applied Logic*, 136, 2005.

[Wil00] Alex Wilkie. On exponentiation – a solution to Tarski's high school algebra problem. *Quaderni di Matematica*, 6, 2000.