Safety, Absoluteness, and Computability

Arnon Avron
School of Computer Science, Tel Aviv University
Tel Aviv, Israel
aa@post.tau.ac.il

Shahar Lev
School of Computer Science, Tel Aviv University
Tel Aviv, Israel
shaharle@post.tau.ac.il

Nissan Levi
School of Computer Science, Tel Aviv University
Tel Aviv, Israel
nisnis.levi@gmail.com

Abstract

The semantic notion of dependent safety is a common generalization of the notion of absoluteness used in set theory and the notion of domain independence used in database theory for characterizing safe queries. This notion has been used in previous works to provide a unified theory of constructions and operations as they are used in different branches of mathematics and computer science, including set theory, computability theory, and database theory. In this paper we provide a complete syntactic characterization of general first-order dependent safety. We also show that this syntactic safety relation can be used for characterizing the set of strictly decidable relations on the natural numbers, as well as for characterizing rudimentary set theory and absoluteness of formulas within it.

2012 ACM Subject Classification Theory of computation → Models of computation

Keywords and phrases Dependent Safety, Computability, Absoluteness, Decidability, Domain Independence

Digital Object Identifier 10.4230/LIPIcs.CSL.2018.8

1 Introduction

The semantic notion of dependent safety is a common generalization of the notion of absoluteness used in set theory ([14, 9]) and the notion of domain independence used in database theory for characterizing safe queries ([1, 20]). It has been introduced in [3] and used there to provide a unified theory of constructions and operations as they are used in different branches of mathematics and computer science, including set theory, computability theory, and database theory. The notion is based on the following two basic ideas (taken from logic programming and database theory):

- From an abstract logical point of view, the focus of a general theory of computations should be on functions of the form:

  $$\lambda y_1, \dots, y_k.\{\langle x_1, \dots, x_n\rangle \in S^n \mid S \models \varphi(x_1, \dots, x_n, y_1, \dots, y_k)\}$$

  where S is a structure for some first-order signature σ, φ is some formula of σ, and $\{\{x_1, \dots, x_n\}, \{y_1, \dots, y_k\}\}$ is a partition of the set $Fv(\varphi)$ of the free variables of φ. Here the tuple $\langle y_1, \dots, y_k\rangle$ provides the input, while the output is the set of answers to the resulting query.

- An allowable query should be *safe* in the sense that the answer to it does not depend on the exact domain of S, but only on the values of the parameters $\{y_1, \dots, y_k\}$ and the part of S which is relevant to them and to the query, under certain conditions concerning the language and the structures that are taken as relevant to the query.

Examples.

1. In every computerized system, what is taken as the type of natural numbers is actually only some finite initial segment of the full set of natural numbers. Therefore a reasonable query should be one that has the same answer in all implementations in which this initial segment includes the inputs to the query and the natural numbers mentioned in it.
2. Query languages for database theory allow only *domain independent* queries, that is: queries for which the corresponding answer would be the same in all databases which have the same scheme and exactly the same tables for it.

The above two principles were translated in [3, 4, 5, 6] into precise definitions. Those works (especially [3]) also naturally lead to the following two theses concerning the development of a general theory of decidability and computability in arbitrary structures:

- The study of decidability of relations should be a part of a more general study of *absoluteness* of formulas and queries;
- The study of computability/constructibility should be a part of a more general study of *dependent safety* of formulas and queries. (We call this type of safety 'dependent' because it is a property of queries which might contain parameters.)

A significant step in this program of developing general theory of dependent safety, absoluteness, and computability was made in [3, 5], where a syntactic framework for these notions was developed. The main virtues of that framework are its generality and universality: it is based on few basic simple syntactic principles, that can be used in what seem to be very different and unrelated areas. The main result of this paper is that it is actually *complete* for general first-order dependent safety and general first-order absoluteness. This explains its generality, and why its principles were independently discovered in different areas. [1]

With the exception of the relatively simple case of databases, the above mentioned general syntactic principles may of course be insufficient in more complex *particular* cases. Still, we show that they suffice also in the case of the arithmetics of the natural numbers, while in the especially important case of set theory our syntactic characterization of absoluteness is equivalent to the usual syntactic approximation that is currently in use.

The structure of the rest of this paper is as follows. Section 2 we review (an improvement of) the framework developed in [3], including all the necessary definitions. In Section 3 we prove the completeness of our syntactic approximation of general first-order dependent safety, while in Section 4 we provide a direct syntactic approximation of general first-order absoluteness. (The latter is a very important special case of the former.) In Section 5 we give a syntactic characterization of absoluteness in the structure $\mathcal{N}$ of the natural numbers. Finally, in Section 6 we study absoluteness in rudimentary set theory, using a language that includes abstract set terms. We show that while the use of such terms involves a proper extension of our syntactic dependent safety, this is not true for syntactic absoluteness.

[1] The principles were originally identified as generalizations of principles used in database theory. As far as we know, this is a rare case in which ideas and principles originally taken from computer science are applied for understanding purely mathematical theories like set theories and number theory.

2 Preliminaries

Throughout this paper, σ is a first-order signature with equality, and *no function symbols* (except for constants). $Fv(\varphi)$ and $Bv(\varphi)$ respectively denote the set of free variables and the set of bound variables of φ. The notation $\varphi(z_1,\ldots,z_k)$ means that $Fv(\varphi)=\{z_1,\ldots,z_k\}$.

2.1 Basic Definitions

▶ **Definition 1.** Let S_1 and S_2 be two structures for σ. S_1 is a *weak substructure* of S_2 (notation: $S_1 \subseteq_\sigma S_2$) if the domain of S_1 is a subset of the domain of S_2, and the interpretations in S_1 and S_2 of the constants of σ are identical.

▶ **Definition 2.** Let $S_1 \subseteq_\sigma S_2$, where S_1 and S_2 are structures for σ. A formula of σ $\varphi(x_1,\ldots,x_n,y_1,\ldots,y_m)$ is safe for S_1 and S_2 with respect to $\{x_1,\ldots,x_n\}$ (notation: $\varphi \succ^{S_1;S_2} \{x_1,\ldots,x_n\}$), if for all $b_1\ldots,b_m \in S_1$:

$$\{\overrightarrow{a} \in S_2^n \mid S_2 \models \varphi(\overrightarrow{a},\overrightarrow{b})\} = \{\overrightarrow{a} \in S_1^n \mid S_1 \models \varphi(\overrightarrow{a},\overrightarrow{b}\})$$

In other words, φ is safe for S_1 and S_2 with respect to $\{x_1,\ldots,x_n\}$ if by viewing $y_1,\ldots,y_m$ as parameters, and assigning elements from S_1 to these parameters, we get a query in $x_1,\ldots,x_n$ having the same answers in S_1 and S_2.

▶ **Definition 3.** A *safety-signature* is a pair (σ,F), where σ is an ordinary first-order signature with equality and no function symbols, and F is a function which assigns to every n-ary predicate symbol of σ a subset of the powerset of $\{1,\ldots,n\}$, so that $F(=)$ is $\{\{1\},\{2\}\}$.

▶ **Definition 4.** Let (σ,F) be a safety-signature, and let S_1, S_2 be structures for σ. S_2 is called a $(\sigma,F)-$*extension* of S_1 (and S_1 is a $(\sigma,F)-$substructure of S_2) if $S_1 \subseteq_\sigma S_2$ and $p(x_1,\ldots,x_n) \succ^{S_1;S_2} \{x_{i_1},\ldots,x_{i_k}\}$ whenever p is an n-ary predicate of σ, $x_1,\ldots,x_n$ are n distinct variables, and $\{i_1,\ldots,i_k\} \in F(p)$.

▶ **Definition 5.** Let (σ,F) be a safety-signature, S a structure for σ, and φ a formula of σ.

1. φ is $(S,F)-$*safe w.r.t.* X (notation: $\varphi \succ_{(S,F)} X$) if $\varphi \succ^{S';S} X$ whenever S is a $(\sigma,F)-$extension of S'. φ is $(S,F)-$*absolute* if $\varphi \succ_{(S,F)} \emptyset$.
2. φ is $(\sigma,F)-$*safe w.r.t.* X ($\varphi \succ_{(\sigma,F)} X$) if it is $(S,F)-$safe w.r.t. X for every structure S for σ. φ is $(\sigma,F)-$*absolute* if $\varphi \succ_{(\sigma,F)} \emptyset$.

▶ **Note 6.** The reason that we have demanded $F(=)$ to be $\{\{1\},\{2\}\}$ (or $\{\{1\},\{2\},\emptyset\}$, which is equivalent) is that $x_1=x_2$ is always safe w.r.t. both $\{x_1\}$ and $\{x_2\}$, but usually not w.r.t. $\{x_1,x_2\}$.

▶ **Note 7.** If $\varphi \succ_{(\sigma,F)} X$ and $Z \subseteq X$, then $\varphi \succ_{(\sigma,F)} Z$. In particular: if $\varphi \succ_{(\sigma,F)} X$ for some X then φ is (σ,F)-absolute. The same applies to $(S,F)-$safety and to $(S,F)-$absoluteness.

▶ **Note 8.** If $F(p)$ is nonempty for every p in σ, then by Note 7 S_1 is a substructure of S_2 (in the usual sense of model theory) whenever S_2 is a (σ,F)-extension of S_1.

2.2 Examples

2.2.1 Computability Theory

Several applications of dependent safety to the theory of computability and decidability have been made in [3]. Here is one of them.

Define the safety-signature $(\sigma_{\mathcal{N}},F_{\mathcal{N}})$ as follows:

- $\sigma_{\mathcal{N}}$ is the first-order signature which includes the constant 0, the binary predicate $\leq$, and the ternary relations P_+, $P_\times$.
- $F_{\mathcal{N}}(\leq) = \{\{1\}\}$, $F_{\mathcal{N}}(P_+) = F_{\mathcal{N}}(P_\times) = \{\emptyset\}$.

The standard structure $\mathcal{N}$ for $\sigma_{\mathcal{N}}$ has the set N of natural numbers as its domain, with the usual interpretations of 0 and $\leq$, and the (graphs of the) operations $+$ and $\times$ on N (viewed as ternary relations on N) as the interpretations of P_+ and $P_\times$, respectively. It is easy to see that $\mathcal{N}$ is a $(\sigma_{\mathcal{N}}, F_{\mathcal{N}})$-extension of a structure S for $\sigma_{\mathcal{N}}$ iff the domain of S is an initial segment of $\mathcal{N}$ (where the interpretations of the relation symbols are the corresponding reductions of the interpretations of those symbols in $\mathcal{N}$). Thus $\varphi \succ_{(\mathcal{N},F_{\mathcal{N}})} X$ iff the query $\{\langle x_1, \ldots, x_n\rangle \in S^n \mid S \models \varphi(x_1, \ldots, x_n, y_1, \ldots, y_k)\}$ is "reasonable" in the sense explained in example 1 above (where $X = \{x_1, \ldots, x_n\}$). Using this observation, it was proved in [3] that a relation R on N is recursively enumerable iff R is definable by a formula of the form $\exists y_1, \ldots, y_n \psi$, where the formula ψ is $(\sigma_{\mathcal{N}}, F_{\mathcal{N}})$-absolute.

2.2.2 Set Theory

Let $\sigma_{ZF} = \{\in\}$, $F_{ZF}(\in) = \{\{1\}\}$. A structure S_2 for σ_{ZF} is a $(\sigma_{ZF}, F_{ZF})-$extension of S_1 iff S_2 is an extension of S_1, and $x_1 \in x_2 \succ^{S_1;S_2} \{x_1\}$. The latter condition means that S_1 is a transitive substructure of S_2. Therefore $\varphi \succ_{(\sigma_{ZF},F_{ZF})} \emptyset$ iff the following holds whenever S_1 is a transitive substructure of S_2: $S_1 \models \varphi \;\Leftrightarrow\; S_2 \models \varphi$. Hence a formula is (σ_{ZF}, F_{ZF})-absolute iff it is absolute in the usual sense of set theory. (See e.g. [14].)

Other applications to set theories of dependent safety in general, and of $\succ_{(\sigma_{ZF},F_{ZF})}$ in particular, have been made in [5] and [4]. In [5] it is suggested that an abstract set term $\{x \mid \varphi\}$ denotes a predicatively acceptable set if $\varphi \succ_{(\sigma_{ZF},F_{ZF})} \{x\}$. In [4] the relation $\succ_{(\sigma_{ZF},F_{ZF})}$ is used as the basis for purely logical characterizations of the comprehension schemas allowed in various set theories (including ZF).

2.2.3 Databases

From a logical point of view, a database of scheme $D = \{P_1, \ldots, P_n\}$ is just a given set of *finite* interpretations of $P_1, \ldots, P_n$. A corresponding query language is usually an ordinary first-order language which is based on a signature σ with equality such that σ contains D, but no function symbols. A query is called *domain independent* ([1, 20]) if its answer is the same in all interpretations in which $P_1, \ldots, \ldots, P_n$ are given by the database, while the interpretations of all other predicate symbols (like $<$ or $\leq$) and of the constants are absolute (and externally given). It can easily be seen that a formula φ is domain independent iff $\varphi \succ_{(\sigma,F)} Fv(\varphi)$ for the function F defined by: $F(Q) = \{\{1, \ldots, n_Q\}\}$ in case $Q \in \{P_1, \ldots, P_n\}$ (where n_Q is the arity of Q), while $F(Q) = \{\emptyset\}$ otherwise.

2.2.4 Querying the Web

In [15] the web is modeled as an ordinary database augmented with three more special relations (together with some other, which for simplicity we ignore): $N(id, title, \ldots)$, $C(node, value)$, $L(source, destination, \ldots)$. The intuitive interpretations of these relations are the following:

- The relation N contains the Web objects which are identified by a Uniform Resource Locator (URL). *id* represents the URL and is a key.
- The meaning of C is that the string which is represented by its second argument occurs within the body of the document in the URL which is represented by its first argument.
- The relation L holds between nodes *source* and *destination* if there is a hypertext link from the first to the second.

The question investigated in [15] is: what queries should be taken as safe, if we assume that what is practically possible in the case of N and L is to list all their tuples which *correspond to a given first argument*, while C is only assumed to be decidable. It is not difficult to see that the notion of safety given there for this framework is equivalent to (σ_{web}, F_{web})-safe in our sense, where $\{L, N, C\} \subseteq \sigma_{web}$, and F is defined like in ordinary databases, except that $F(L) = \{2, , \dots, , m\}$ (where m is the arity of L), $F(N) = \{2, , \dots, , k\}$ (where k is the arity of N), and $F(C) = \{\emptyset\}$.

2.3 The Corresponding Syntactic Relation

In [10] it was proved that the property of domain independence in databases is undecidable. In [3] it was shown that the property of (σ, F)-absoluteness is also in general undecidable. This means that in order to use the relation $\succ_{(\sigma,F)}$ in practice we need a decidable syntactic approximation. The one that was used in [3, 4, 5] is presented in the next definition. It was inspired by the recursive definition of syntactic safety given in [20], and generalizes it in a sense explained below.[2]

▶ **Definition 9.** Given a safety-signature (σ, F), we recursively define the relation $\succ^s_{(\sigma,F)}$ between formulas of σ and sets of variables as follows:

1. $p(t_1, \dots, t_n) \succ^s_{(\sigma,F)} X$ in case p is an n-ary predicate symbol of σ, and there is $I \in F(p)$ such that:
   a. For every $x \in X$ there is $i \in I$ such that $x = t_i$.
   b. $X \cap Fv(t_j) = \emptyset$ for every $j \in \{1, \dots, n\} \backslash I$.
2. $\neg\varphi \succ^s_{(\sigma,F)} \emptyset$ if $\varphi \succ^s_{(\sigma,F)} \emptyset$.
3. $\varphi \vee \psi \succ^s_{(\sigma,F)} X$ if $\varphi \succ^s_{(\sigma,F)} X$ and $\psi \succ^s_{(\sigma,F)} X$
4. $\varphi \wedge \psi \succ^s_{(\sigma,F)} X \cup Y$ if $\varphi \succ^s_{(\sigma,F)} X$, $\psi \succ^s_{(\sigma,F)} Y$, and either $Fv(\varphi) \cap Y = \emptyset$ or $Fv(\psi) \cap X = \emptyset$.
5. $\exists y.\varphi \succ^s_{(\sigma,F)} X \backslash \{y\}$ if $y \in X$ and $\varphi \succ^s_{(\sigma,F)} X$.

▶ **Theorem 10** ([3]). *$\succ^s_{(\sigma,F)}$ is sound: if $\varphi \succ^s_{(\sigma,F)} \{x_1, \dots, x_n\}$ then $\varphi \succ_{(\sigma,F)} \{x_1, \dots, x_n\}$*

▶ **Note 11.** In what follows $\forall x_1 \dots \forall x_n.\varphi \to \psi$ as an abbreviation for $\neg\exists x_1 \dots \exists x_n.\varphi \wedge \neg\psi$. Using items 2,4, and 5 from Definition 9, this implies that $\forall x_1 \dots \forall x_n.\varphi \to \psi \succ^s_{(\sigma,F)} \emptyset$ if $\varphi \succ^s_{(\sigma,F)} \{x_1, \dots, x_n\}$ and $\psi \succ^s_{(\sigma,F)} \emptyset$. We shall use this fact freely.

▶ **Note 12.** It follows from Definition 9 and the fact that $F(=)$ is $\{\{1\}, \{2\}\}$ that $x = t \succ^s_{(\sigma,F)} \{x\}$ and $t = x \succ^s_{(\sigma,F)} \{x\}$ in case $x \notin Fv(t)$, and $t = s \succ^s_{(\sigma,F)} \emptyset$ for every t, s.

Examples.

1. The set of formulas φ such that $\varphi \succ^s_{(\sigma_\mathcal{N}, F_\mathcal{N})} \emptyset$ includes all formulas in the well-known set of arithmetical Δ_0-formulas (also called "bounded formulas" or "σ_0-formulas" in [17]). In the context of $\sigma_\mathcal{N}$ these are the formulas in which all quantifications are of the form $\exists x \leq y$ (or $\forall x \leq y$, by Note 11), where x and y are distinct variables.
2. Similarly, the set of formulas φ such that $\varphi \succ^s_{(\sigma_{ZF}, F_{ZF})} \emptyset$ is an extension of the set of set-theoretical Δ_0 formulas ([14]).[3] However, in this case not only this special case of syntactic dependent safety is important. In fact, if $\varphi(x_1, \dots, x_n, y_1, \dots, y_k) \succ^s_{(\sigma_{ZF}, F_{ZF})} \{x_1, \dots, x_n\}$ then the function $\lambda y_1, \dots, y_k.\{\langle x_1, \dots, x_n\rangle \mid \varphi\}$ is rudimentary. (Rudimentary functions

[2] Other closely related works in database theory are e.g. [16], [19], and [18].
[3] In the context of σ_{ZF} Δ_0-formulas (again also called "bounded formulas") are the formulas in which all quantifications are of the form $\exists x \in y$ (or $\forall x \in y$, by Note 11), where x and y are variables.

were independently introduced by Gandy in [12] and by Jensen in [13]. See also [9].) In particular: if $\varphi(x_1,\ldots,x_n,y_1,\ldots,y_k) \succ^s_{(\sigma_{ZF},F_{ZF})} \{x_1,\ldots,x_n\}$ then the function $\lambda y_1 \in \mathcal{HF},\ldots,y_k \in \mathcal{HF}.\{\langle x_1,\ldots,x_n\rangle \in \mathcal{HF}^n \mid \mathcal{HF} \models \varphi(x_1,\ldots,x_n,y_1,\ldots,y_k)\}$ (where $\mathcal{HF}$ is the set of hereditarily finite sets) is a computable function from $\mathcal{HF}^k$ to $\mathcal{HF}$. (We shall return to this example in Section 6.)

3. Let D, σ, and F be like in Section 2.2.3. Then $\varphi \succ^s_{(\sigma,F)} Fv(\varphi)$ for any formula φ which is syntactically safe according to the definition in [20].
4. $\varphi \succ^s_{(\sigma_{web},F_{web})} Fv(\varphi)$ for any formula φ which is safe according to the "Safe Web Calculus" given in [15] as a syntactic approximation for the class of (σ_{web},F_{web})-safe formulas.

▶ **Note 13.** It is easy to see that if $\varphi \succ^s_{(\sigma,F)} X$ and $Y \subseteq X$ then $\varphi \succ^s_{(\sigma,F)} Y$. In particular, if $\varphi \succ^s_{(\sigma,F)} X$ then $\varphi \succ^s_{(\sigma,F)} \emptyset$.

3 The General Completeness Theorem

Our main goal in this section is to prove an appropriate converse to Theorem 10.

▶ **Notation 14.** $\varphi \equiv \psi$ if φ and ψ are logically equivalent, and $Fv(\varphi) = Fv(\psi)$.

▶ **Lemma 15.** *Let (σ,F) be a safety-signature. Let φ and ψ be two formulas of σ such that $Y \subseteq Fv(\varphi) \cap Fv(\psi)$. If φ and ψ are logically equivalent, then $\varphi \succ_{(\sigma,F)} Y$ iff $\psi \succ_{(\sigma,F)} Y$. In particular: if $\varphi \equiv \psi$ then for every Y it holds that $\varphi \succ_{(\sigma,F)} Y$ iff $\psi \succ_{(\sigma,F)} Y$.*

Proof. Immediate from the definitions. ◀

▶ **Theorem 16.** *Let (σ,F) be a safety-signature such that σ includes a constant. Then for every φ and Y, $\varphi \succ_{(\sigma,F)} Y$ iff there exists ψ such that $\psi \succ^s_{(\sigma,F)} Y$ and $\varphi \equiv \psi$.*

Proof. We begin with some notations. If S is a structure for σ, and v is an assignment in S, then $S,v \models \varphi$ denotes that φ is satisfied in S by the assignment v. $T \vdash^t \varphi$ denotes that $S,v \models \varphi$ whenever $S,v \models \psi$ for every $\psi \in T$. If $\bar{x} = \langle x_1,\ldots,x_m\rangle$ is a finite list of distinct variables, and $\bar{a} \in S^m$, then we denote by $\bar{x} := \bar{a}$ some assignment v in S such that $v(x_i) = a_i$ for every $1 \le i \le m$. If $Fv(\varphi) = \{x_1,\ldots,x_m\}$ and $\bar{a} \in S^m$, then $S \models \varphi(\bar{a})$ means that $S, \bar{x} := \bar{a} \models \varphi$.

Let (σ,F) be a safety-signature.

▶ **Lemma 17.** *Let $(\widehat{\sigma},\widehat{F})$ be the safety-signature such that $\widehat{\sigma}$ is σ without the predicates p for which $F(p) = \emptyset$ and $\widehat{F}$ is the restriction of F to predicates of $\widehat{\sigma}$. If $\varphi \succ_{(\sigma,F)} Y$ then there exists a formula $\widehat{\varphi}$ of $\widehat{\sigma}$ such that $\widehat{\varphi} \succ_{(\widehat{\sigma},\widehat{F})} Y$ and $\varphi \equiv \widehat{\varphi}$.*

Proof. Let S_1 and S_2 be two structures for σ that have the same domain and the same interpretations for the constants of σ and the predicates of $\widehat{\sigma}$. Then S_1 and S_2 are (σ,F)-substructures of one another, and so $S_1,v \models \varphi$ iff $S_2,v \models \varphi$ for every assignment v in their common domain. Therefore Beth definability theorem implies that there exists a formula $\widehat{\varphi}$ of $\widehat{\sigma}$ such that $\varphi \equiv \widehat{\varphi}$. By Lemma 15, $\widehat{\varphi} \succ_{(\sigma,F)} Y$ and so $\widehat{\varphi} \succ_{(\widehat{\sigma},\widehat{F})} Y$. ◀

▶ **Lemma 18.** *Let S_1,S_2,S_3 be structures for σ such that S_1 is a substructure of S_2, S_2 is a substructure of S_3, and S_1 is a (σ,F)-substructure of S_3. Then S_1 is a (σ,F)-substructure of S_2.*

Proof. Let p be a n-ary predicate of σ, and let $I \in F(p)$. Suppose that $a_1,\ldots,a_n \in S_2$ and $a_i \in S_1$ for every $i \in \{1,\ldots,n\}\backslash I$.

- Assume $S_2 \models p(\bar{a})$. Since S_2 is a substructure of S_3 then $S_3 \models p(\bar{a})$. Since S_1 is a (σ, F)-substructure of S_3, $\bar{a} \in S_1^n$ and $S_1 \models p(\bar{a})$.
- Assume $\bar{a} \in S_1^n$ and $S_1 \models p(\bar{a})$. Since S_1 is a substructure of S_2 then $S_2 \models p(\bar{a})$. ◀

▶ **Lemma 19.** *For a structure S for σ and $\bar{a} \in S^m$, let $\alpha_{(\sigma,F)}[S, \bar{a}]$ be the substructure of S whose domain is the set of all $b \in S$ for which there exists a formula $\theta(x_1, \dots, x_m, z)$ of σ (where $x_1, \dots, x_m, z$ are $m+1$ distinct variables) such that $\theta(\bar{x}, z) \succ^s_{(\sigma,F)} \{z\}$ and $S \models \theta(\bar{a}, b)$. Then $\alpha_{(\sigma,F)}[S, \bar{a}]$ is a (σ, F)-substructure of S.*

Proof. First note that $\alpha_{(\sigma,F)}[S, \bar{a}]$ is indeed a well-defined substructure of S. This follows from the facts that σ contains no function symbols, and that for every constant c in σ, the formula $c = z$ of σ satisfies $c = z \succ^s_{(\sigma,F)} \{z\}$, assuring that $\alpha_{(\sigma,F)}[S, \bar{a}]$ contains all the interpretations in S of the constants of σ. (In particular: $\alpha_{(\sigma,F)}[S, \bar{a}] \neq \emptyset$.)[4]

Now suppose that p is a n-ary predicate of σ, $I \in F(p)$, $\bar{b} \in S^n$, and $b_j \in \alpha_{(\sigma,F)}[S, \bar{a}]$ for every $j \in \{1, \dots, n\} \backslash I$.

- Assume $b_i \in \alpha_{(\sigma,F)}[S, \bar{a}]$ for every $i \in I$ and $\alpha_{(\sigma,F)}[S, \bar{a}] \models p(\bar{b})$. Since $\alpha_{(\sigma,F)}[S, \bar{a}]$ is a substructure of S, we get that $S \models p(\bar{b})$.
- Assume $S \models p(\bar{b})$. Let $x_1, \dots, x_m, y_1, \dots, y_n, z$ be $m + n + 1$ distinct variables. Let $j \in \{1, \dots, n\} \backslash I$. Since we assume that $b_j \in \alpha_{(\sigma,F)}[S, \bar{a}]$, the definition of $\alpha_{(\sigma,F)}[S, \bar{a}]$ implies that there exists a formula $\theta_j(\bar{x}, y_j)$ of σ such that $\theta_j(\bar{x}, y_j) \succ^s_{(\sigma,F)} \{y_j\}$ and $S \models \theta_j(\bar{a}, b_j)$. Define the following formulas of σ:

$$\xi(\bar{x}, \bar{y}) : \left(\bigwedge_{j \in \{1, \dots, n\} \backslash I} \theta_j(\bar{x}, y_j) \right) \wedge p(\bar{y})$$

$$\mu_i(\bar{x}, z) : \exists \bar{y} [\xi(\bar{x}, \bar{y}) \wedge z = y_i] \quad (i \in I)$$

Now we know that:

$$\bigwedge_{j \in \{1, \dots, n\} \backslash I} \theta_j(\bar{x}, y_j) \succ^s_{(\sigma,F)} \{y_j \mid j \in \{1, \dots, n\} \backslash I\}$$

$$p(\bar{y}) \succ^s_{(\sigma,F)} \{y_i \mid i \in I\}$$

It follows that $\xi(\bar{x}, \bar{y}) \succ^s_{(\sigma,F)} \{y_1, \dots y_n\}$. Moreover, $S \models \xi(\bar{a}, \bar{b})$. Thus $\mu_i(\bar{x}, z) \succ^s_{(\sigma,F)} \{z\}$ and $S \models \mu_i(\bar{a}, b_i)$ for every $i \in I$. By definition of $\alpha_{(\sigma,F)}[S, \bar{a}]$, $b_i \in \alpha_{(\sigma,F)}$ for every $i \in I$. Since $\alpha_{(\sigma,F)}[S, \bar{a}]$ is a substructure of S then $\alpha_{(\sigma,F)}[S, \bar{a}] \models p(\bar{b})$. ◀

▶ **Definition 20.** Let φ and $\psi(x_1, \dots, x_m, z)$ be two formulas of σ such that φ contains no bound instances of $x_1, \dots, x_m$. $Re_{\psi(\bar{x},z)}[\varphi]$ is the formula obtained by recursively replacing in φ all subformulas of the forms $\exists w \theta$ with $\exists w. \psi(\bar{x}, w) \wedge \theta$.

▶ **Lemma 21.** *Assume that $F(p) \neq \emptyset$ for every predicate p of σ. Let φ and $\psi(x_1, \dots, x_m, z)$ be two formulas of σ such that φ contains no bound instances of $x_1, \dots, x_m$, and $\psi(\bar{x}, z) \succ^s_{(\sigma,F)} \{z\}$. Then $Re_{\psi(\bar{x},z)}[\varphi] \succ^s_{(\sigma,F)} \emptyset$.*

Proof. The proof is by induction on the structure of φ:

1. Since $F(p) \neq \emptyset$ for every predicate p of σ, $\theta \succ^s_{(\sigma,F)} \emptyset$ for every atomic formula θ of σ (see Note 7).

[4] This is the place in the proof of Theorem 16 where we use the assumption that σ includes a constant.

2. By clauses 3,4,5 of Definition 9, $\theta \succ^s_{(\sigma,F)} \emptyset$ for every boolean combination θ of formulas $\theta_1,\dots,\theta_k$ of σ such that $\theta_i \succ^s_{(\sigma,F)} \emptyset$ for every $1 \le i \le k$.
3. By clause 6 of Definition 9 $\exists w.\psi(\bar{x},w) \wedge \theta \succ^s_{(\sigma,F)} \emptyset$ whenever $\theta \succ^s_{(\sigma,F)} \emptyset$. ◀

End of the proof of Theorem 16

By Theorem 10 and Lemma 15, it suffices to prove that if $\varphi \succ_{(\sigma,F)} Y$ then there exists a formula ψ of σ such that $\psi \succ^s_{(\sigma,F)} Y$ and $\varphi \equiv \psi$. Moreover, by Lemma 17 we may assume that σ contains no predicate p for which $F(p) = \emptyset$.

Let $x_1,\dots,x_m,y_1,\dots,y_n,z$ be $m+n+1$ distinct variables, and let $\varphi(\bar{x},\bar{y})$ be a formula of σ such that $\varphi(\bar{x},\bar{y}) \succ_{(\sigma,F)} \{y_1,\dots,y_n\}$. Without a loss in generality, we may assume that φ contains no bound instances of $x_1,\dots,x_m$. Let σ_q be obtained from σ by the addition of a new $(m+1)$-ary predicate symbol q. Define in σ_q:

$$\psi'(\bar{x},\bar{y}) \ := \ Re_{q(\bar{x},z)}[\varphi(\bar{x},\bar{y})] \wedge \bigwedge_{i=1}^{n} q(\bar{x},y_i)$$

Let T be the set of all formulas of σ_q of the form $\forall z[\theta(\bar{x},z) \to q(\bar{x},z)]$ where $\theta(\bar{x},z) \succ^s_{(\sigma,F)} \{z\}$ (and so θ is in σ). We will prove that $T \vdash^t \forall\bar{y}(\varphi(\bar{x},\bar{y}) \leftrightarrow \psi'(\bar{x},\bar{y}))$.

Let S be a structure for σ_q and let $\bar{a}$ be a tuple in S^m such that $S,\, \bar{x} := \bar{a} \models T$. Let S_3 be the structure for σ obtained from S by restricting it to σ. Let S_2 be the substructure of S_3 whose domain is the set of all $b \in S$ such that $S \models q(\bar{a},b)$. Let S_1 be the structure $\alpha_{(\sigma,F)}[S_3,\bar{a}]$. By definition of T and the fact that $S,\, \bar{x} := \bar{a} \models T$, S_1 is a substructure of S_2. By Lemmas 19 and 18, S_1 is a (σ,F)-substructure of both S_2 and S_3. In addition, $\varphi(\bar{x},\bar{y}) \succ_{(\sigma,F)} \{y_1,\dots,y_n\}$ and $\bar{a} \in S_1^m$. (The latter can be justified by the fact that for every $1 \le i \le m$, the formula $x_i = z$ of σ satisfies $x_i = z \succ^s_{(\sigma,F)} \{z\}$.) Therefore, for every $\bar{b} \in S_3^n$:

$$S_3 \models \varphi(\bar{a},\bar{b}) \iff \bar{b} \in S_1^n \wedge S_1 \models \varphi(\bar{a},\bar{b})$$

$$S_3 \models \varphi(\bar{a},\bar{b}) \iff \bar{b} \in S_2^n \wedge S_2 \models \varphi(\bar{a},\bar{b})$$

Since $\varphi(\bar{x},\bar{y})$ is a formula of σ (since it does not contain the predicate q), we get that for every $\bar{b} \in S_3^n$, $S \models \varphi(\bar{a},\bar{b})$ iff $S_3 \models \varphi(\bar{a},\bar{b})$. By relativization and definition of S_2, we get that for every $\bar{b} \in S_3^n$, $\bar{b} \in S_2^n \wedge S_2 \models \varphi(\bar{a},\bar{b})$ iff $S \models \psi'(\bar{a},\bar{b})$. By transitivity, we get that for every $\bar{b} \in S_n^3$, $S \models \varphi(\bar{a},\bar{b})$ iff $S \models \psi'(\bar{a},\bar{b})$. Because S_3 and S have the same domain, we get that $S,\, \bar{x} := \bar{a} \models \forall\bar{y}(\varphi(\bar{x},\bar{y}) \leftrightarrow \psi'(\bar{x},\bar{y}))$.

We proved that $T \vdash^t \forall\bar{y}(\varphi(\bar{x},\bar{y}) \leftrightarrow \psi'(\bar{x},\bar{y}))$. By compactness, there exists a finite subset T_1 of T such that:

$$(*) \quad T_1 \vdash^t \forall\bar{y}(\varphi(\bar{x},\bar{y}) \leftrightarrow \psi'(\bar{x},\bar{y}))$$

Suppose $T_1 = \{\forall z[\theta_i(\bar{x},z) \to q(\bar{x},z)] \mid 1 \le i \le n\}$, and let $\mu(\bar{x},z)$ be the disjunction of $\theta_1(\bar{x},z),\dots,\theta_n(\bar{x},z)$. Then μ is a formula of σ such that $\mu(\bar{x},z) \succ^s_{(\sigma,F)} \{z\}$. Obtain the set of formulas T_2 of σ and the formula $\psi(\bar{x},\bar{y})$ of σ from T_1 and ψ' respectively by replacing every atom of the form $q(\bar{x},z)$ in them by $\mu(\bar{x},z)$. Since classical first-order logic is structural (that is: its consequence relation is closed under allowed substitutions of formulas for predicates symbols), $(*)$ implies that $T_2 \vdash^t \forall\bar{y}(\varphi(\bar{x},\bar{y}) \leftrightarrow \psi(\bar{x},\bar{y}))$. Since the definition of μ entails that all formulas in T_2 are logically valid, this implies that $\varphi \equiv \psi$. Moreover, $\psi(\bar{x},\bar{y})$ is $Re_{\mu(\bar{x},z)}[\varphi(\bar{x},\bar{y})] \wedge \bigwedge_{i=1}^{n} \mu(\bar{x},y_i)$. Hence Lemma 21 entails that $\psi(\bar{x},\bar{y}) \succ^s_{(\sigma,F)} \{y_1,\dots,y_n\}$ (relying on earlier assumption that $F(p) \neq \emptyset$ for every predicate p in σ). ◀

▶ **Note 22.** Theorem 16 is not always correct as is in case σ contains no constant. Take for example the case where σ is empty. (So the language has '$=$' as its sole predicate symbol, and no constants or function symbols.) It is easy to prove that there is no formula ψ of this language such that $\psi \succ^s_{(\sigma,F)} Fv(\psi)$. Hence there is no ψ in this language such that $\psi \equiv x \neq x$ and $\psi \succ^s_{(\sigma,F)} \{x\}$, even though obviously $x \neq x \succ_{(\sigma,F)} \{x\}$ (where $F(=)$ is $\{\{1\},\{2\}\}$). Still, $x \neq x$ *is* logically equivalent to some formula ψ such that $\psi \succ^s_{(\sigma,F)} \{x\}$ and $x \in Fv(\psi)$ (e.g. $\psi := x = y \wedge x \neq x$). It is indeed easy to infer from Theorem 16 that in general, if σ contains no constant and $\varphi \succ_{(\sigma,F)} X$ then there is a formula ψ such that ψ is logically equivalent to φ, $\psi \succ^s_{(\sigma,F)} \{x\}$, and $Fv(\varphi) \subseteq Fv(\psi)$.

4 Characterization of General Absoluteness

As we saw in the first section, while in database theory the main interest is in formulas which are domain-independent (i.e. formulas which are safe with respect to their full set of free variables), in formal number theory (and in computability theory) and in set theory the main interest has been in absolute formulas. [5] Now in the previous section we have given a general syntactic characterization of absoluteness: Given a safety signature (σ, F), a formula φ is (σ, F)-absolute iff there exists a formulas ψ such that $\varphi \equiv \psi$, and $\varphi \succ^s_{(\sigma,F)} \emptyset$. However, this characterization of the *property* of absoluteness is based in an essential way on the *relation* $\succ^s_{(\sigma,F)}$ between formulas and sets of variables. Therefore in order to check whether a certain formula φ is absolute using this characterization, one should check on the way with respect to what sets of variables are the subformulas of φ safe. In contrast, in formal number theory and in set theory a direct syntactic approximation of absoluteness has been used in the form of what is called in both Δ_0-formulas. In this section we generalize the notion of Δ_0-formulas to arbitrary safety signatures, and use the generalized notion for providing a *direct* syntactic characterization of (σ, F)-absoluteness. Note that in order to use this characterization one needs not know anything about the more general binary relation $\succ^s_{(\sigma,F)}$.

▶ **Notation 23.** For a formula φ and a set of variables $Z = \{z_1, \ldots, z_k\}$, $\exists^Z.\varphi$ denotes the formula $\exists z_1, \ldots \exists z_k.\varphi$, and $\forall^Z.\varphi$ denotes the formula $\forall z_1, \ldots \forall z_k.\varphi$.

▶ **Definition 24.** Let (σ, F) be a safety signature. The class $\Delta_{(\sigma,F)}$ of formulas[6] is recursively defined as follows:

1. $p(t_1, \ldots, t_n) \in \Delta_{(\sigma,F)}$ in case p is an n-ary predicate symbol of σ, and $F(p) \neq \emptyset$.
2. If $\varphi, \psi \in \Delta_{(\sigma,F)}$ then so is any boolean combination of them.
3. $\exists^Z.\varphi_1 \wedge \varphi_2 \in \Delta_{(\sigma,F)}$ and $\forall^Z.\varphi_1 \rightarrow \varphi_2 \in \Delta_{(\sigma,F)}$ in case $\varphi_2 \in \Delta_{(\sigma,F)}$, $\varphi_1 = p(t_1, \ldots, t_n)$, where p is an n-ary predicate of σ other than $=$, and $\varphi_1 \succ^s_{(\sigma,F)} Z$, that is: there is $I \in F(p)$ such that:
   a. For every $z \in Z$ there is $i \in I$ such that $z = t_i$.
   b. $Z \cap Fv(t_j) = \emptyset$ for every $j \in \{1, \ldots, n\} \backslash I$.

Examples

$\Delta_{(\sigma_{ZF},F_{ZF})}$ is exactly the class Δ_0 used in set theory. Similarly, $\Delta_{(\sigma_{\mathcal{N}},F_{\mathcal{N}})}$ is equivalent to the class Δ_0 used in formal number theory. (See the first two examples in Section 2.3.)

[5] Actually, absolute formulas may be of interest for databases too, since they can be used for effectively decidable yes-or-no queries. See [3].

[6] This is a proper extension of the class GF of guarded formulas ([2]), in case F is the particular function which assigns the powerset of $\{1, ..., n\}$ to every n-ary primitive predicate R of σ.

▶ **Theorem 25.** *$\varphi \succ^{s}_{(\sigma,F)} \emptyset$ iff there exists a formula $\varphi' \in \Delta_{(\sigma,F)}$ such that $\varphi \equiv \varphi'$.*

Proof. Obviously, if $\varphi' \in \Delta_{(\sigma,F)}$ then $\varphi' \succ^{s}_{(\sigma,F)} \emptyset$. Hence the condition is sufficient. In order to prove that it is also necessary, we need the following lemma:

▶ **Lemma 26.** *Let $\succ^{*}_{(\sigma,F)}$ be defined like $\succ^{s}_{(\sigma,F)}$, except that the clause for conjunction is replaced by:*

If $\varphi_1 \succ^{*}_{(\sigma,F)} Y_1$, $\varphi_2 \succ^{*}_{(\sigma,F)} Y_2$, $Fv(\varphi_1) \cap Y_2 = \emptyset$ and φ_1 is an atomic formula or $Y_1 = \emptyset$, then $\varphi_1 \wedge \varphi_2 \succ^{*}_{(\sigma,F)} Y_1 \cup Y_2$.

Then for every formula φ, $\varphi \succ^{s}_{(\sigma,F)} Y$ iff there is a formula φ' such that $\varphi' \succ^{*}_{(\sigma,F)} Y$ and $\varphi \equiv \varphi'$.

Proof. Obviously, if $\varphi' \succ^{*}_{(\sigma,F)} Y$ then $\varphi' \succ^{s}_{(\sigma,F)} Y$. Hence the condition is sufficient. In order to prove that it is also necessary, it suffices to show that up to logical equivalence, $\succ^{*}_{(\sigma,F)}$ abides the condition concerning $\wedge$ used in the definition of $\succ^{s}_{(\sigma,F)}$. So assume e.g. that $\varphi_1 \succ^{*}_{(\sigma,F)} Y_1$, $\varphi_2 \succ^{*}_{(\sigma,F)} Y_2$ and $Fv(\varphi_1) \cap Y_2 = \emptyset$. We prove the existence of a formula φ' such that $\varphi_1 \wedge \varphi_2 \equiv \varphi'$ and $\varphi' \succ^{*}_{(\sigma,F)} Y_1 \cup Y_2$. The proof is by induction on the structure of φ_1:

- Assume φ_1 is an atomic formula. Then $\varphi_1 \wedge \varphi_2 \succ^{*}_{(\sigma,F)} Y_1 \cup Y_2$ by the new conjunction safety clause.
- Assume φ_1 is the formula $\psi_1 \vee \psi_2$ where $\psi_1 \succ^{*}_{(\sigma,F)} Y_1$ and $\psi_2 \succ^{*}_{(\sigma,F)} Y_1$. Since $Fv(\varphi_1) = Fv(\psi_1) \cup Fv(\psi_2)$, we know that $Fv(\psi_1) \cap Y_2 = Fv(\psi_2) \cap Y_2 = \emptyset$. Then, by induction assumption, there exist formulas θ_1 and θ_2 such that $\psi_1 \wedge \varphi_2 \equiv \theta_1$, $\psi_2 \wedge \varphi_2 \equiv \theta_2$, $\theta_1 \succ^{*}_{(\sigma,F)} Y_1 \cup Y_2$ and $\theta_2 \succ^{*}_{(\sigma,F)} Y_1 \cup Y_2$. Therefore $\varphi_1 \wedge \varphi_2 \equiv \theta_1 \vee \theta_2$ and $\theta_1 \vee \theta_2 \succ^{*}_{(\sigma,F)} Y_1 \cup Y_2$.
- Assume φ_1 is the formula $\psi_1 \wedge \psi_2$ where $\psi_1 \succ^{*}_{(\sigma,F)} Z_1$, $\psi_2 \succ^{*}_{(\sigma,F)} Z_2$, $Fv(\psi_1) \cap Z_2 = \emptyset$, $Y_1 = Z_1 \cup Z_2$ and ψ_1 is an atomic formula or $Z_1 = \emptyset$, Since $Fv(\psi_2) \cap Y_2 = \emptyset$, we get by induction assumption the existence of a formula θ such that $\psi_2 \wedge \varphi_2 \equiv \theta$ and $\theta \succ^{*}_{(\sigma,F)} Z_2 \cup Y_2$. Since $Fv(\psi_1) \cap (Z_2 \cup Y_2) = \emptyset$, we get that $\varphi_1 \wedge \varphi_2 \equiv \psi_1 \wedge \theta$ and $\psi \wedge \theta \succ^{*}_{(\sigma,F)} Y_1 \cup Y_2$.
- Assume φ_1 is the formula $\neg\psi$ where $\psi \succ^{*}_{(\sigma,F)} \emptyset$. Then $Y_1 = \emptyset$ and then $\varphi_1 \wedge \varphi_2 \succ^{*}_{(\sigma,F)} Y_1 \cup Y_2$ by the new conjunction safety clause.
- Assume $\varphi_1 = \exists z\psi$ where $\psi \succ^{*}_{(\sigma,F)} Y_1 \cup \{z\}$ and $z \notin Y_1$. In addition, assume w.l.o.g. that $z \notin Fv(\varphi_2)$. Since $Fv(\psi) \cap Y_2 = \emptyset$, we get by induction assumption the existence of a formula θ such that $\psi \wedge \varphi_2 \equiv \theta$ and $\theta \succ^{*}_{(\sigma,F)} Y_1 \cup Y_2 \cup \{z\}$. Then $\varphi_1 \wedge \varphi_2 \equiv \exists z\theta$ and $\exists z\theta \succ^{*}_{(\sigma,F)} Y_1 \cup Y_2$.

This completes the induction. ◀

End of the proof of Theorem 25

We show the necessity of the condition by proving a stronger claim: For every formula φ such that $\varphi \succ^{s}_{(\sigma,F)} Y$ there exists a formula $\varphi' \in \Delta_{(\sigma,F)}$ such that $\exists^{Y}\varphi \equiv \varphi'$. By Lemma 26, we only need to prove the latter under the assumption that $\varphi \succ^{*}_{(\sigma,F)} Y$. The proof in this case is by induction on the structure of φ:

- Assume φ is atomic. If $Y = \emptyset$ then $\varphi \in \Delta_{(\sigma,F)}$. Otherwise, choosing $y \in Y$, we get $y = y \succ_{(\sigma,F)} \emptyset$, $\exists^{Y}(\varphi \wedge y = y) \in \Delta_{(\sigma,F)}$ and $\exists^{Y}\varphi \equiv \exists^{Y}(\varphi \wedge y = y)$.
- Assume φ is the formula $\psi_1 \vee \psi_2$ where $\psi_1 \succ^{*}_{(\sigma,F)} Y$ and $\psi_2 \succ^{*}_{(\sigma,F)} Y$. By induction assumption, there exists formulas $\theta_1 \in \Delta_{(\sigma,F)}$ and $\theta_2 \in \Delta_{(\sigma,F)}$ such that $\exists^{Y}\psi_1 \equiv \theta_1$ and $\exists^{Y}\psi_2 \equiv \theta_2$. Then $\theta_1 \vee \theta_2 \in \Delta_{(\sigma,F)}$ and $\exists^{Y}\varphi \equiv \theta_1 \vee \theta_2$.

- Assume φ is $\psi_1 \wedge \psi_2$ where $\psi_1 \succ^*_{(\sigma,F)} Y_1$, $\psi_2 \succ^*_{(\sigma,F)} Y_2$, $Fv(\psi_1) \cap Y_2 = \emptyset$, $Y = Y_1 \cup Y_2$ and ψ_1 is an atomic formula or $Y_1 = \emptyset$. By induction assumption, there exists a formula $\theta_2 \in \Delta_{(\sigma,F)}$ such that $\exists^{Y_2}\psi_2 \equiv \theta_2$. If $Y_1 = \emptyset$ then, by induction assumption, there exists a formula $\theta_1 \in \Delta_{(\sigma,F)}$ such that $\psi_1 \equiv \theta_1$ and so $\theta_1 \wedge \theta_2 \in \Delta_{(\sigma,F)}$ and $\exists^Y \varphi \equiv \theta_1 \wedge \theta_2$. Otherwise, if $Y_1 \neq \emptyset$ then ψ_1 is an atomic formula, $\exists^{Y_1}(\psi_1 \wedge \theta_2) \in \Delta_{(\sigma,F)}$ and $\exists^Y \varphi \equiv \exists^{Y_1}(\psi_1 \wedge \theta_2)$.
- Assume φ is the formula $\neg\psi$ where $\psi \succ^*_{(\sigma,F)} \emptyset$. Then $Y = \emptyset$ and, by induction assumption, there exists a formula $\theta \in \Delta_{(\sigma,F)}$ such that $\psi \equiv \theta$ and so $\neg\theta \in \Delta_{(\sigma,F)}$ and $\varphi \equiv \neg\theta$.
- Assume $\varphi = \exists z\psi$ where $\psi \succ^*_{(\sigma,F)} Y \cup \{z\}$ and $z \notin Y$. By induction assumption, there exists a formula $\theta \in \Delta_{(\sigma,F)}$ such that $\exists^Y \varphi \equiv \exists^{Y\cup\{z\}}.\psi \equiv \theta$.

By Note 11, this completes the proof. ◀

5 Characterization of Absoluteness in $\mathcal{N}$

Theorem 25 is about *general* (σ, F)- absoluteness, and so it is not applicable to the notion of (S, F)-absoluteness, where S is a structure for σ. In this section we prove a similar theorem for one particular, but very important, case of (S, F)-absoluteness: $(\mathcal{N}, \sigma_\mathcal{N})$-absoluteness.

▶ **Note 27.** Recall that in Section 2.2.1 it was noted that by a result of [3], relation R on N is recursively enumerable iff R is definable by a formula of the form $\exists y_1, \ldots, y_n\psi$, where the formula ψ is $(\sigma_\mathcal{N}, F_\mathcal{N})$-absolute. It was further observed there that every relation on $\mathcal{N}$ that is defined by a $(\mathcal{N}, F_\mathcal{N})$-absolute formula is decidable, and that there are decidable relations on $\mathcal{N}$ that are not definable by any formula φ such that $\varphi \succ^s_{(\sigma_\mathcal{N},F_\mathcal{N})} \emptyset$. It was left open whether every decidable relation on $\mathcal{N}$ is definable by a $(\sigma_\mathcal{N}, F_\mathcal{N})$-absolute formula, and whether every relation which is definable by such a formula is already definable by a formula φ such that $\varphi \succ^s_{(\sigma_\mathcal{N},F_\mathcal{N})} \emptyset$. In view of the above-mentioned observations, the next theorem implies that the answer to the first question is negative, while the answer to the second is positive.

▶ **Theorem 28.** *A formula φ such that $Fv(\varphi) \neq \emptyset$ is $(\mathcal{N}, \sigma_\mathcal{N})$-absolute iff there is an arithmetical bounded formula*[7] *φ' such that φ is equivalent in $\mathcal{N}$ to φ'.*

Proof. We assume without loss of generality that for every formula ψ it holds that $Fv(\psi) \cap Bv(\psi) = \emptyset$, and that any two variables that appear in ψ to the right of two different occurrences of quantifiers are different. For $k \in N$ we denote by $\mathcal{N}_k$ the structure with domain $\{0, 1, \ldots, k\}$, and the interpretations of the relation symbols are the corresponding reductions of the interpretations of those symbols in $\mathcal{N}$. For an assignment v, a variable u, and a natural value n, we denote $v[u := n]$ the assignment that agree with v on all variables except u, and assigns the value n to u.

Given a formula φ and a set $\{x_1, \ldots, x_k\}$ of variables such that $\{x_1, \ldots, x_k\} \cap Bv(\varphi) = \emptyset$, we denote by $\varphi^{\leq x_1,\ldots,x_k}$ the formula $Re_{\psi(\bar{x},z)}[\varphi]$ (Definition 20), where $\psi(\bar{x}, z)$ is $z \leq x_1 \vee \ldots \vee z \leq x_k$. The proof of the theorem is based on the following three lemmas:

▶ **Lemma 29.** *$\varphi^{\leq x_1,\ldots,x_k}$ is logically equivalent to a bounded formula for every formula φ.*

Proof. This follows immediately from the definitions, and the fact that $\exists z.(z \leq x_1 \vee \ldots \vee z \leq x_k) \wedge \psi$ is logically equivalent to the formula $\exists z \leq x_1.\psi \wedge \ldots \wedge \exists z \leq x_k.\psi$. ◀

[7] See first example in Section 2.3.

▶ **Lemma 30.** *Let φ be a formula, let $\{y, x_1, \ldots, x_k\}$ be a set of variables* s.t. $Bv(\varphi) \cap \{y, x_1, \ldots, x_k\} = \emptyset$, *and let v be an assignment* s.t. $v(y) \leq \max(v(x_1), \ldots, v(x_k))$. *Then the following holds*

$$\mathcal{N}, v \vDash \varphi^{\leq x_1,\ldots,x_k} \quad \textit{iff} \quad \mathcal{N}, v \vDash \varphi^{\leq y,x_1,\ldots,x_k}$$

Proof. We prove it by a structural induction on φ. The only non-trivial case is when φ is of the form $\exists z.\psi$. In this case $\varphi^{\leq x_1,\ldots,x_k}$ is $\exists z.(z \leq x_1 \vee \ldots \vee z \leq x_k) \wedge \psi^{\leq x_1,\ldots,x_k}$. Hence $\mathcal{N}, v \vDash \varphi^{\leq x_1,\ldots,x_k}$ iff (*) there exists $n \in N$ such that:

$$\mathcal{N}, v[z := n] \vDash (z \leq x_1 \vee \ldots \vee z \leq x_k) \wedge \psi^{\leq x_1,\ldots,x_k}$$

Obviously, $v'(y) \leq \max(v'(x_1), \ldots, v'(x_k))$ for every assignment v' that agrees with v on $\{y, x_1, \ldots, x_k\}$. Hence the induction hypothesis for ψ implies that for any such v':

$$\mathcal{N}, v' \vDash \psi^{\leq x_1,\ldots,x_k} \quad \text{iff} \quad \mathcal{N}, v' \vDash \psi^{\leq y,x_1,\ldots,x_k}. \tag{1}$$

Also for any such v', $\mathcal{N}, v' \vDash z \leq y \vee z \leq x_1 \vee \ldots \vee z \leq x_k$ iff $\mathcal{N}, v' \vDash z \leq x_1 \vee \ldots \vee z \leq x_k$ (because $z \notin Bv(\varphi)$, and so $z \notin \{y, x_1, \ldots, x_n\}$). This observation and 1 imply that (*) holds iff there exists $n \in N$ such that:

$$\mathcal{N}, v[z := n] \vDash (z \leq y \vee z \leq x_1 \vee \ldots \vee z \leq x_k) \wedge \psi^{\leq y,x_1,\ldots,x_k}$$

And this is equivalent to: $\mathcal{N}, v \vDash \varphi^{\leq y,x_1,\ldots,x_k}$. ◀

▶ **Lemma 31.** *Let $\{x_1, \ldots, x_k\}$ be a non-empty set of variables, let φ be a formula such that $Fv(\varphi) \subseteq \{x_1, \ldots, x_k\}$, and let v be an assignment. Denote by $\tilde{m} := \max(v(x_1), \ldots, v(x_k))$. Then:*

$$\mathcal{N}_{\tilde{m}}, v \vDash \varphi \quad \textit{iff} \quad \mathcal{N}, v \vDash \varphi^{\leq x_1,\ldots,x_k} \tag{2}$$

Proof. By a structural induction on φ. Again the only non-trivial case is when φ is of the form $\exists z.\psi$. So let v be an assignment, and assume that $\mathcal{N}_{\tilde{m}}, v \vDash \exists y.\psi$. It follows that there exists $n \in N$, $0 \leq n \leq \tilde{m}$, s.t. $\mathcal{N}_{\tilde{m}}, v[y := n] \vDash \psi$. By the induction hypothesis for ψ and $\{y, x_1, \ldots, x_k\}$, it holds that $\mathcal{N}, v[y := n] \vDash \psi^{\leq y,x_1,\ldots,x_k}$. Denote the assignment $v[y := n]$ by v'. Since $v'(y) = n \leq \tilde{m} = \max(v'(x_1), \ldots, v'(x_k))$, $\mathcal{N}, v[y := n] \vDash \psi^{\leq x_1,\ldots,x_k}$ by Lemma 30. Hence $\mathcal{N}, v \vDash \exists y.(y \leq x_1 \vee \ldots \vee y \leq x_k) \wedge \psi^{\leq x_1,\ldots,x_k}$, that is: $\mathcal{N}, v \vDash \varphi^{\leq x_1,\ldots,x_k}$. To prove the converse we just repeat the argument in reverse order: Assume that $\mathcal{N}, v \vDash \varphi^{\leq x_1,\ldots,x_k}$. This means that $\mathcal{N}, v \vDash \exists y.(y \leq x_1 \vee \ldots \vee y \leq x_k) \wedge \psi^{\leq x_1,\ldots,x_k}$. It follows that there is $0 \leq n \leq \tilde{m}$ s.t. $\mathcal{N}, v[y := n] \vDash \psi^{\leq x_1,\ldots,x_k}$. Using Lemma 30, it follows that $\mathcal{N}, v[y := n] \vDash \psi^{\leq y,x_1,\ldots,x_k}$. Therefore the induction hypothesis and the fact that $\tilde{m} := \max(v(x_1), \ldots, v(x_k))$ together imply that $\mathcal{N}_{\tilde{m}}, v[y := n] \vDash \psi$. Hence $\mathcal{N}_{\tilde{m}}, v \vDash \varphi$. ◀

End of the proof of Theorem 28

Suppose that $\varphi \succ_{(\mathcal{N},F_{\mathcal{N}})} \emptyset$, and let $Fv(\varphi) = \{x_1, \ldots, x_k\}$ where $k \geq 1$. Consider the formula $\varphi' = \varphi^{\leq x_1,\ldots,x_k}$. ($\varphi' \succ^s_{\mathcal{N}} \emptyset$ by Lemma 29.) We show that

$$\{\bar{n} \in N^k \mid \mathcal{N},\ \bar{x} := \bar{n} \vDash \varphi\} = \{\bar{n} \in N^k \mid \mathcal{N},\ \bar{x} := \bar{n} \vDash \varphi^{\leq x_1,\ldots,x_k}\}$$

Let $\langle n_1, \ldots, n_k \rangle \in N^k$, and let v be an assignment that assigns n_i to x_i for every $1 \leq i \leq k$. Since $\varphi \succ_{(\mathcal{N},F_{\mathcal{N}})} \emptyset$, $\mathcal{N}, v \vDash \varphi$ iff $\mathcal{N}_{\max(n_1,\ldots,n_k)}, v \vDash \varphi$. By Lemma 31 $\mathcal{N}_{\max(n_1,\ldots,n_k)}, v \vDash \varphi$ iff $\mathcal{N}, v \vDash \varphi^{\leq x_1,\ldots,x_k}$, and the claim follows. ◀

6 Absoluteness in Rudimentary Set Theory

To complete the picture concerning absoluteness, we return in this section to the area in which this notion has first been introduced: set theory. In Sections 2.2.2 and 2.3 (second example) we have noted that the notion of (σ_{ZF}, F_{ZF})-absoluteness is identical to Gödel's original notion of absoluteness, and that $\{\varphi \mid \varphi \succ^{s}_{(\sigma_{ZF},F_{ZF})} \emptyset\}$ is a natural extension of the set of Δ_0-formulas in the language of σ_{ZF}. However, in order to fully exploit the power of the idea of dependent safety in the framework of set theory, we need to use a language which is stronger (and more natural) than the official language of ZF. The main feature of the stronger language, $\mathcal{L}_{RST}$, is that it employs a rich class of set terms of the form $\{x \mid \varphi\}$. Of course, not every formula φ can be used in such a term. The basic idea in [5] was that from a predicative point of view, one should allow only formulas which are safe with respect to $\{x\}$. Since safety is a semantic notion, again what is used instead in [5] is a formal approximation $\succ_{RST}$. $\succ_{RST}$ is basically the natural extension of $\succ^{s}_{(\sigma_{ZF},F_{ZF})}$ to the richer language. However, the definition of that very language depends in turn on that of $\succ_{RST}$. Accordingly, the sets of terms and formulas of $\mathcal{L}_{RST}$, and the relation $\succ_{RST}$, are defined together by a simultaneous induction:

▶ **Definition 32.** The language $\mathcal{L}_{RST}$ is defined as follows:

Terms:

1. Every variable is a term.
2. If x is a variable, and φ is a formula such that $\varphi \succ_{RST} \{x\}$, then $\{x \mid \varphi\}$ is a term (and $Fv(\{x \mid \varphi\}) = Fv(\varphi) - \{x\}$).

Formulas:

1. If t, s are terms than $t = s$ and $t \in s$ are atomic formulas.
2. If φ and ψ are formulas, then $\neg\varphi$, $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$, and $\exists x\varphi$ are formulas.

The safety relation $\succ_{RST}$:

1. $\varphi \succ_{RST} \emptyset$ if φ is atomic.
2. $\varphi \succ_{RST} \{x\}$ if $\varphi \in \{x \in x, x = t, t = x, x \in t\}$, and $x \notin Fv(t)$.
3. $\neg\varphi \succ_{RST} \emptyset$ if $\varphi \succ_{RST} \emptyset$.
4. $\varphi \vee \psi \succ_{RST} X$ if $\varphi \succ_{RST} X$ and $\psi \succ_{RST} X$.
5. $\varphi \wedge \psi \succ_{RST} X \cup Y$ if $\varphi \succ_{RST} X$, $\psi \succ_{RST} Y$, and $Y \cap Fv(\varphi) = \emptyset$ or $X \cap Fv(\psi) = \emptyset$.
6. $\exists y\varphi \succ_{RST} X - \{y\}$ if $y \in X$ and $\varphi \succ_{RST} X$.

▶ **Theorem 33** ([5]). *Every term of $\mathcal{L}_{RST}$ with n free variables explicitly defines an n-ary rudimentary function, and every rudimentary function is defined by some term of $\mathcal{L}_{RST}$.*

The two most basic formal set theories in the language $\mathcal{L}_{RST}$ are described next.

▶ **Definition 34.**

1. RST^m is the first-order theory with equality in the language $\mathcal{L}_{RST}$[8] which has the following axioms:
   - Extensionality: $\forall z(z \in x \leftrightarrow z \in y) \to x = y$
   - Comprehension: $\forall x(x \in \{x \mid \varphi\} \leftrightarrow \varphi)$ if $\varphi \succ_{RST} \{x\}$.
2. RST is the system obtained from RST^m by the addition of the following schema:
   - $\in$-induction: $(\forall x(\forall y(y \in x \to \varphi\{y/x\}) \to \varphi)) \to \forall x\varphi$

[8] $\mathcal{L}_{RST}$ has richer classes of terms than those allowed in orthodox first-order systems. In particular: a variable can be bound in them within a term. The notion of a term being free for substitution should be extended accordingly. Otherwise the rules/axioms concerning the quantifiers, terms, and equality remain unchanged.

▶ **Note 35.** The use of $\in$-induction seems to be predicatively justified. Therefore RST is the basic system used in [5]. However, for the results below we use just one very weak corollary of it: $\forall x.x \notin x$. (It is needed for the new clause $x \in x \succ_{RST} \{x\}$ in Definition 32.)

▶ **Note 36.** RST (or even just RST^m) serves in [5], [6] and [7] as the basis of *computational set theories*. By this we mean a theory whose set of closed terms suffices for defining its minimal model, and can be used to make explicit the potential computational content of set theories (first suggested and partially demonstrated in [8]). On the other hand, such theories also suffice (as is shown in [6] and [7]) for developing large portions of what was called by Feferman in [11] 'scientifically applicable mathematics'.

▶ **Note 37.** Despite the fact that the definition of $\succ_{RST}$ uses almost exactly the same principles that underlie that of $\succ^s_{(\sigma_{ZF},F_{ZF})}$ (with the slight addition that $x \in x \succ_{RST} \{x\}$, while we only have $x \in x \succ^s_{(\sigma_{ZF},F_{ZF})} \emptyset$), the use of abstract set terms induces a significantly stronger safety relation on the basic language of σ_{ZF}. The reason is that the fact that $x = t \succ_{RST} \{x\}$ is equivalent in RST^m to the following principle:

- If $\varphi \succ_{RST} \{y\}$ then $\forall y(y \in x \leftrightarrow \varphi) \succ_{RST} \{x\}$ if $x \notin Fv(\varphi)$.

(It is not difficult to show that the addition of this clause indeed suffices for getting a system in the language of ZF which is equivalent to RST.) Nevertheless, the next theorem and its corollary imply that when it comes to *absoluteness*, the addition of the abstract set terms does not provide extra expressive power.

▶ **Theorem 38.** *Let ψ be a Δ_0 formula of σ_{ZF} (that is, without abstract set terms).*

1. *If x is a variable, and t is a term which is free for x in ψ, then $\psi\{t/x\}$ is equivalent in RST to a Δ_0-formula of σ_{ZF}.*
2. *If $\varphi \succ_{RST} \{x_1, \dots, x_n\}$ then the formula $\exists x_1 \dots x_n(\varphi \wedge \psi)$ is equivalent in RST to a Δ_0-formula of σ_{ZF}.*

Proof. By a simultaneous induction on the complexity of t and φ.

- If t is a variable then the claim is obvious.
- Suppose t is $\{y \mid \varphi\}$, where $\varphi \succ_{RST} \{y\}$. We prove the claim for t by an internal induction on the complexity of ψ.
  - If x is not free in ψ then the claim is obvious.
  - If ψ is $x \in x$ then $\psi\{t/x\}$ is equivalent in RST to the formula $\exists x \in x.x \in x$.
  - If ψ is $x = x$ then $\psi\{t/x\}$ is equivalent in RST to the formula $\neg\exists x \in x.x \in x$.
  - Suppose ψ is $z \in x$, where z is different from x. We may assume that z is not bound in φ. Then $\psi\{t/x\}$ is equivalent in RST to $\varphi\{z/y\}$. Since $\varphi \succ_{RST} \emptyset$, φ is equivalent in RST to a Δ_0-formula by the induction hypothesis. Hence so does $\varphi\{z/y\}$.
  - Suppose ψ is $z = x$ or $x = z$, where z is a variable different from x. We may assume that z is not y. Then $\psi\{t/x\}$ is equivalent in RST to $(\forall y \in z.\varphi) \wedge \neg\exists y(\varphi \wedge y \notin z)$. Since $\varphi \succ_{RST} \{y\}$ and $\varphi \succ_{RST} \emptyset$, φ and $\exists y(\varphi \wedge y \notin z)$ are equivalent in RST to Δ_0-formulas by the external induction hypothesis for φ. It follows that so is $\psi\{t/x\}$.
  - Suppose ψ is $x \in z$, where z is a variable different from x. Let w be a fresh variable. Then $\psi\{t/x\}$ is logically equivalent to $\exists w \in z.w = t$. By the previous case, $w = t$ is equivalent in RST to a Δ_0 formula. Hence so is $\psi\{t/x\}$.
  - If ψ is $\neg\psi_1$ or $\psi_1 \wedge \psi_2$, or $\psi_1 \vee \psi_2$, then the claim for ψ follows from the induction hypothesis for ψ_1 and ψ_2.
  - If ψ is of the form $\exists z \in w.\psi_1$, where both w and z are different from x, then the claim for ψ is immediate from the internal induction hypothesis for ψ_1.

- Suppose ψ is of the form $\exists z \in x.\psi_1$ (where z is different from x). Since t is free for x in ψ, z does not occur free in φ, and we may assume that it does not occur in φ at all. Then $\psi\{t/x\}$ is equivalent in RST to $\exists z(\varphi\{z/y\} \wedge \psi_1\{t/x\})$. Since z does not occur in φ and $\varphi \succ_{RST} \{y\}$, also $\varphi\{z/y\} \succ_{RST} \{z\}$. Hence by the external induction hypothesis for φ and the internal induction hypothesis for ψ_1, $\psi\{t/x\}$ is equivalent in RST to a Δ_0 formula.

- Suppose φ is atomic (and so $\varphi \succ_{RST} \emptyset$). Then φ is either $t_1 \in t_2$ or $t_1 = t_2$ for some terms t_1 and t_2. Since $x \in y$ and $x = y$ are Δ_0-formulas, it follows by applying the induction hypotheses for t_1 and t_2 that φ is equivalent to a Δ_0-formula. Hence $\varphi \wedge \psi$ is equivalent to a Δ_0-formula whenever ψ is.
- Suppose that φ is of the form $x \in x$, where x is a variable, Then $\varphi \succ_{RST} \{x\}$, and so we have to prove that $\exists x \in x.\psi$ is equivalent to a Δ_0-formula. This is obvious.
- Suppose that φ is of the form $x \in t$, where $x \notin Fv(t)$. Since $\varphi \succ_{RST} \{x\}$ in this case, we have to prove that for every Δ_0-formula ψ, $\exists x(x \in t \wedge \psi)$ is equivalent to a Δ_0-formula. This follows from the induction hypothesis for t, since the last formula is $\theta\{t/z\}$, where z is a fresh variable, and θ is the Δ_0-formula $\exists x(x \in z) \wedge \psi$ (note that since $x \notin Fv(t)$, t is free for z in θ).
- Suppose that φ is of the form $x = t$ or $t = x$, where x is not free in t. Since $\varphi \succ_{RST} \{x\}$ in this case, we have to prove that for every Δ_0-formula ψ, $\exists x(x = t \wedge \psi)$ is equivalent to a Δ_0-formula. By changing bound variables, we may assume that t is free for x in ψ. This and the fact that $x \notin Fv(t)$ together imply that $\exists x(x = t \wedge \psi)$ is logically equivalent to $\psi\{t/x\}$. This formula, in turn, is equivalent in RST to a Δ_0-formula by our induction hypothesis for t.
- Suppose φ is $\neg\varphi_1$, where $\varphi_1 \succ_{RST} \emptyset$ (and so $\neg\varphi_1 \succ_{RST} \emptyset$). By induction hypothesis for φ, φ is equivalent in RST to a Δ_0-formula. Hence so is $\neg\varphi \wedge \psi$ for every Δ_0-formula ψ.
- Suppose φ is $\varphi_1 \vee \varphi_2$, where $\varphi_1 \succ_{RST} \{x_1, \dots, x_n\}$ and $\varphi_2 \succ_{RST} \{x_1, \dots, x_n\}$ (and so $\varphi \succ_{RST} \{x_1, \dots, x_n\}$). Then $\exists x_1 \dots x_k(\varphi \wedge \psi)$ is logically equivalent to $\exists x_1 \dots x_k(\varphi_1 \wedge \psi) \vee \exists x_1 \dots x_k(\varphi_2 \wedge \psi)$. Hence the induction hypothesis for φ_1 and φ_2 entails that $\exists x_1 \dots x_k(\varphi \wedge \psi)$ is equivalent in RST to a Δ_0-formula whenever ψ is.
- Suppose φ is $\varphi_1 \wedge \varphi_2$, $\varphi_1 \succ_{RST} \{x_1, \dots, x_n\}$, $\varphi_2 \succ_{RST} \{y_1, \dots, y_k\}$, $\{y_1, \dots, y_k\} \cap Fv(\varphi_1) = \emptyset$ (so $\varphi \succ_{RST} \{x_1, \dots, x_n, y_1, \dots, y_k\}$). Then $\exists x_1 \dots x_n y_1 \dots y_k(\varphi \wedge \psi)$ is equivalent to $\exists x_1 \dots x_n(\varphi_1 \wedge \exists y_1 \dots y_k(\varphi_2 \wedge \psi))$. By applying the induction hypothesis twice, we get that $\exists x_1 \dots y_k(\varphi \wedge \psi)$ is equivalent in RST to a Δ_0-formula whenever ψ is.
- Suppose φ is $\exists y\varphi_1$ where $\varphi_1 \succ_{RST} \{x_1, \dots, x_n, y\}$. Let ψ be a Δ_0-formula. Then $\exists x_1 \dots x_n(\varphi \wedge \psi)$ is logically equivalent to the formula $\exists x_1 \dots x_n z(\varphi_1\{z/y\} \wedge \psi)$, where z is a fresh variable. Since $\varphi_1\{z/y\} \succ_{RST} \{x_1, \dots, x_n, z\}$, the induction hypothesis implies that $\exists x_1 \dots x_n z(\varphi_1\{z/y\} \wedge \psi)$ is equivalent in RST to a Δ_0-formula. Hence so is $\exists x_1 \dots x_n(\varphi \wedge \psi)$. ◀

▶ **Corollary 39.** *If $\varphi \succ_{RST} \emptyset$ then φ is equivalent in RST to a Δ_0-formula of σ_{ZF}.*

▶ **Note 40.** On the other hand, if $X \neq \emptyset$, then it can happen that $\varphi \succ_{RST} X$, but $\theta \not\succ_{RST} X$, where θ is the Δ_0-formula to which φ is equivalent according to the construction given in the last proof. Thus if φ is $x = \{y\}$, then θ is the $y \in x \wedge \forall z \in x.z = y$, so $\theta \not\succ_{RST} \{x\}$, even though $\varphi \succ_{RST} \{x\}$. This problem cannot be solved by adding to the definition of $\succ_{RST}$ the clause mentioned in Note 37, because $\forall y(y \in x \leftrightarrow \varphi)$ is not necessarily a Δ_0-formula in case φ is. From the above theorem it follows that it is equivalent in RST to a Δ_0-formula θ, but then again there seems to be no guarantee that $\theta \succ_{RST} \{x\}$.

7 Conclusion and Further Research

We have shown that the syntactic framework developed in [3, 5] for the semantic notions of dependent safety and absoluteness is complete in the case of *general* first-order logic in languages without function symbols. Therefore it promises to be rather adequate for the general theory of constructibility, decidability, and computability envisaged in [3]. The next stages of this research program will involve the following goals:

1. Extending the general theory of dependent safety for languages with function symbols.
2. The completeness result given in this paper is with respect to the class of *all* structures for a given signature. However, frequently we are mainly interested only with a subclass of that class. two particularly important cases for which an extension of the general theory developed here is needed are:
   a. The class of finite models.
   b. The class of the models of some given theory.
3. For computability theory we might need to restrict our attention to *specific* central structures. Thus in section (5) we characterized the absolute formulas of the important structure $(\mathcal{N}, F_{\mathcal{N}})$. It is not clear whether the same can be done for other basic important structures, like the structure of hereditarily finite sets $\mathcal{HF} = (\mathbb{HF}, \langle\in\rangle)$ (where $\in$ has its usual meaning, and $x \in y$ is safe with respect to $\{x\}$).
4. Providing concrete applications of our results in specific areas. This includes:
   - Database theory (e.g. Datalog extended with arithmetic).
   - MKM (Mathematical Knowledge Management), in particular: the formalization of scientifically applicable mathematics in a type-free, predicative setting ([5]).

References

1 S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases.* Addison-Wesley, 1995.

2 H. Andréka, J. van Benthem, and I. Németi. Modal languages and bounded fragments of predicate logic. *Journal of Philosophical Logic*, 27:217–274, 1998.

3 A. Avron. Constructibility and decidability versus domain independence and absoluteness. *Theoretical Computer Science*, 394:144–158, 2008.

4 A. Avron. A framework for formalizing set theories based on the use of static set terms. In A. Avron, N. Dershowitz, and A. Rabinovich, editors, *Pillars of Computer Science*, volume 4800 of *LNCS*, pages 87–106. Springer, 2008.

5 A. Avron. A new approach to predicative set theory. In R. Schindler, editor, *Ways of Proof Theory*, onto series in mathematical logic, pages 31–63. onto verlag, 2010.

6 A. Avron and L. Cohen. Formalizing scientifically applicable mathematics in a definitional framework. *Journal of Formalized Reasoning*, 9(1):53–70, 2016.

7 A. Avron and L. Cohen. A minimal computational theory of a minimal computational universe. In *Proc. of LFCS 2018*, pages 37–54, 2018.

8 D. Cantone, E. Omodeo, and A. Policriti. *Set theory for computing: from decision procedures to declarative programming with sets.* Springer, 2001.

9 K. J. Devlin. *Constructibility.* Perspectives in Mathematical Logic. Springer-Verlag, 1984.

10 R. A. Di Paola. The recursive unsolvability of the decision problem for the class of definite formulas. *J. ACM*, 16:324–327, 1969.

11 S. Feferman. Why a little bit goes a long way: Logical foundations of scientifically applicable mathematics. In *PSA: Proceedings of the Biennial Meeting of the Philosophy of Science Association*, pages 442–455, 1992.

12 R. O. Gandy. Set-theoretic functions for elementary syntax. In *Axiomatic set theory, Part 2*, pages 103–126. AMS, Providence, Rhode Island, 1974.

13 R. B. Jensen. The fine structure of the constructible hierarchy. *Annals of Mathematical Logic*, 4:229–308, 1972.

14 K. Kunen. *Set Theory, An Introduction to Independence Proofs*. North-Holland, 1980.

15 A. O. Mendelzon and T. Milo. Formal models of web queries. In *Proceedings of the Sixteenth ACM Symposium on Principles of Database Systems*, pages 134–143, 1997.

16 R. Ramakrishnan, F. Bancilhon, and A. Silberschatz. Safety of recursive horn clauses with infinite relations. In *ACM SIGACT-SIGMOD Symp. on Principles of Database Systems, San Diego*, 1987.

17 R. M. Smullyan. *The Incompleteness Theorems*. Oxford University Press, 1992.

18 R. Topor. Safe database queries with arithmetic relations. In *Proceedings of the 14th Australian Computer Science Conference*, pages 1–13, Sydney, 1991.

19 Rodney W. Topor. Domain-independent formulas and databases. *Theoretical Computer Science*, 52(3):281–306, 1987.

20 J.D. Ullman. *Principles of Database and Knowledge-base Systems*. Computer Science Press, 1988.