Theoretical Computer Science 935 (2022) 21–46

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Unprovability results for clause set cycles

Stefan Hetzl *, Jannik Vierling

Vienna University of Technology, Institute of Discrete Mathematics and Geometry, Austria

ARTICLE INFO

Article history:
Received 5 October 2021
Received in revised form 28 June 2022
Accepted 8 July 2022
Available online 13 July 2022
Communicated by O. Arieli

Keywords:
Automated inductive theorem proving
Cyclic proofs
Weak arithmetical theories

ABSTRACT

The notion of clause set cycle abstracts a family of methods for automated inductive theorem proving based on the detection of cyclic dependencies between clause sets. By discerning the underlying logical features of clause set cycles, we are able to characterize clause set cycles by a logical theory. We make use of this characterization to provide practically relevant unprovability results for clause set cycles that exploit different logical features.

1. Introduction

The subject of automated inductive theorem proving (AITP) is a subfield of automated theorem proving, that aims at automating the process of finding proofs that involve mathematical induction. The most prominent application of automated inductive theorem proving is the formal verification of hardware and software. Another field of application of automated inductive theorem proving is the formalization of mathematical statements, where AITP systems assist humans in formalizing statements by discharging lemmas automatically, suggest inductions [25], or explore the theory [21,33].

Finding a proof by mathematical induction essentially amounts to finding suitable induction formulas [19]. This is a challenging task, because induction formulas have in general a higher syntactic complexity than the formula one wants to prove. This phenomenon is commonly known as the *non-analyticity* of induction formulas and can for example manifest itself in the number of free variables as well as the number of quantifier alternations of the induction formula. Indeed, in the language of primitive recursive arithmetic there is a sequence of quantifier-free formulas whose proofs require induction formulas of unbounded quantifier complexity. We refer to [19] for a precise exposition of the non-analyticity phenomenon.

A large variety of methods for automating mathematical induction has been developed. Methods usually differ in the type of induction formulas they generate, the calculus they are integrated in, and other more technical parameters such as the degree of automation, the input encoding, semantics of datatypes, and so on. For example there are methods based on term rewriting [27], theory exploration [9], and integration into saturation-based provers [23,22], [11,12], [13], [29,15].

The current methodology in automated inductive theorem proving focuses on empirical evaluations of its methods. A given method is usually evaluated on a set of benchmark problems such as [10]. Such an evaluation provides evidence about the strengths and weaknesses of a method but does not result in a systematic understanding of the underlying principles. In particular, it is difficult to compare the methods with each other in terms of their logical strength and to provide explanations of the failures of a given method.

* Corresponding author.
E-mail addresses: stefan.hetzl@tuwien.ac.at (S. Hetzl), jannik.vierling@tuwien.ac.at (J. Vierling).

https://doi.org/10.1016/j.tcs.2022.07.003

The work in this article is part of a research program that addresses this problem by formally analyzing AITP systems in order to discern their underlying logical principles. The analysis of an AITP system typically begins by developing a suitable abstraction. After that, the abstraction is simulated by a logical theory, whose properties can be investigated by applying powerful results and techniques from mathematical logic. Analyzing families of methods in the uniform formalism of logic allows us not only to understand the strength of individual methods but also to compare the methods with each other. Furthermore, approximating an AITP system by a logical theory is a prerequisite to providing concrete and practically meaningful unprovability results. These results are especially valuable because they allow us to determine the logical features that a given method lacks. Thus, negative results drive the development of new and more powerful methods.

In [18] the authors of this article have introduced the notion of clause set cycle as an abstraction of the n-clause calculus [23,22]—an extension of the superposition calculus by a cycle detection mechanism. In particular, we have shown an upper bound on the strength of clause set cycles in terms of induction for $\exists_1$ formulas and moreover that this bound is optimal with respect to the quantifier complexity of induction formulas.

In this article we continue this analysis of clause set cycles. By discerning the logical features underlying the formalism of clause set cycles more precisely, we are able to provide an exact characterization of refutation by a clause set cycle in terms of a logical theory. After that, we make use of the characterization of clause set cycles to provide practically meaningful clause sets that are not refutable by clause set cycles, but that are refutable by induction on quantifier-free formulas. Hence, the results in this article settle in particular Conjecture 4.7 of [18]. We provide unrefutability results that exploit different logical features of clause set cycles. This allows to recognize features that are particularly restrictive.

In Section 2 we will first introduce general notions and results about the logical setting that we use in this article. In Section 3 we carry out the analysis of clause set cycles which culminates in Section 3.3 with two unprovability results for clause set cycles. The two unprovability results exploit different logical features of clause set cycles. The proof of the first unprovability result is straightforward, whereas the second unprovability result relies on a more involved independence result in the setting of linear arithmetic whose proof is carried out in Section 4.

2. Preliminary definitions

In this section we introduce some definitions that we will use throughout the article. In Section 2.1 we will briefly describe the logical formalism. In Section 2.2 we describe the setting of formal linear arithmetic, in which we will formulate in Section 3.3 a family of clause sets that are refutable by open induction but that are not refutable by clause set cycles.

2.1. Formulas, theories, and clauses

We work in a setting of classical logic with equality, that is, the logic provides besides the usual logical symbols a binary infix predicate symbol $=$ representing equality. A first-order language L is a set of predicate symbols and function symbols together with their respective arities. Let S be a predicate or function symbol, then we write S/n to indicate that S has arity n. Terms, atoms, and formulas are constructed as usual from function symbols, variable symbols, the logical connectives $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$, and the quantifiers $\exists$ and $\forall$. A ground term is a term that does not contain variables. The set of all L formulas is denoted by $\mathcal{F}(L)$. A sentence is a formula that does not contain free variables. Let $x_1, \ldots, x_n$ be variables, $t_1, \ldots, t_n$ terms, and φ a formula, then $\varphi[x_1/t_1, \ldots, x_n/t_n]$ denotes the simultaneous substitution of x_i by t_i for $i = 1, \ldots, n$ in φ.

In this article we are more interested in the axioms of a theory, rather than the deductive closure of these axioms. Hence, we define a theory as a set axioms and manipulate the deductive closure by means of the first-order provability relation.

Definition 1 *(Theories and provability).* A theory T is a set of sentences called the axioms of T. Let T, U be theories, then by $T + U$ we denote the theory axiomatized by $T \cup U$. Let φ be a formula, then we write $T \vdash \varphi$ if φ is provable in first-order logic from the axioms T. Let Γ be a set of formulas, then we write $T \vdash \Gamma$ if $T \vdash \gamma$ for each $\gamma \in \Gamma$. Furthermore, we write $T \equiv U$, if $T \vdash U$ and $U \vdash T$.

Let T be a theory and φ a formula, then we write $T + \varphi$ to denote the theory axiomatized by the axioms of T and the universal closure of φ. In this article we will be particularly interested in formulas with a restricted number of quantifier alternations.

Definition 2. We say that a formula φ (possibly containing free variables) is $\forall_0$ or $\exists_0$ if φ is quantifier-free. Moreover, we say that a formula $\varphi(\vec{z})$ is $\forall_{k+1}$ ($\exists_{k+1}$) if it is of the form $(\forall \vec{x})\psi(\vec{x}, \vec{z})$ ($(\exists \vec{x})\psi(\vec{x}, \vec{z})$) and ψ is $\exists_k$ ($\forall_k$). Let L be a first-order language, then by $\mathrm{Open}(L)$, $\exists_k(L)$, and $\forall_k(L)$, we denote the quantifier-free formulas, $\exists_k$ formulas, and the $\forall_k$ formulas of the language L. A theory is said to be $\exists_k$ ($\forall_k$) if all of its axioms are $\exists_k$ ($\forall_k$) sentences.

Clause sets are an alternative representation of $\forall_1$ formulas, that is preferred by automated theorem provers because of its uniformity.

Definition 3 *(Literals, clauses, clause sets).* Let L be a first-order language. By an L literal we understand an L atom or the negation of an L atom. An L clause is a finite set of L literals. An L clause set is a set of L clauses. Whenever the language L is clear from the context, we simply speak of atoms, literals, clauses and clause sets.

We will now recall some basic model-theoretic concepts.

Definition 4. Let L be a first-order language, then L structures and the first-order satisfaction relation $\models$ are defined as usual. Let $L' \subseteq L$ be a first-order language and M an L structure, then by $M|_{L'}$ we denote the L' reduct of M. Let M be an L structure, then we write $b \in M$ to express that b is an element of the domain of M. Formulas and clauses are interpreted as usual. In particular, a clause is interpreted as the universal closure of the disjunction of its literals. Let Δ be a set of L formulas and L clauses, then $M \models \Delta$ if $M \models \delta$ for each $\delta \in \Delta$.

Let us conclude this section by introducing some notation to manipulate clauses and clause sets.

Definition 5. By cls we denote a fixed function that assigns to every $\forall_1$ sentence φ, a clause set $cls(\varphi)$ over the language of φ such that φ and $cls(\varphi)$ are logically equivalent. Let Γ be a set of $\forall_1$ sentences, then we define $cls(\Gamma) := \bigcup_{\gamma \in \Gamma} cls(\gamma)$. Furthermore, by cls^{-1} we denote a fixed function that assigns to every clause set $\mathcal{C}$ a $\forall_1$ sentence $cls^{-1}(\mathcal{C})$ over the language of $\mathcal{C}$ such that $\mathcal{C}$ and $cls^{-1}(\mathcal{C})$ are logically equivalent.

Lemma 6. *Let $\mathfrak{C}$ be a finite set of clause sets, then there exists a clause set $\mathcal{C}'$ such that $M \models \mathcal{C}'$ if and only if there exists $\mathcal{C} \in \mathfrak{C}$ such that $M \models \mathcal{C}$.*

Proof. Let $\mathfrak{C} = \{\mathcal{C}_1, \dots, \mathcal{C}_n\}$, now let $\psi := \bigvee_{i=1}^{n} cls^{-1}(\mathcal{C}_i)$. Then ψ is logically equivalent to a $\forall_1$ sentence ψ'. Now we define $\mathcal{C}' := cls(\psi')$. It is clear that $M \models \mathcal{C}'$ if and only if there exists $i \in \{1, \dots, n\}$ such that $M \models \mathcal{C}_i$. □

2.2. Induction and formal arithmetic

In this section we introduce some basic notions about induction and formal arithmetic. In particular we introduce the setting of linear arithmetic in which we formulate an unrefutability result for clause set cycles in Section 3.3.

Inductive theorem provers customarily work in a many-sorted setting with a notion of inductive datatypes encompassing at least the natural numbers, lists, trees, and sometimes even more complicated types such as mutually recursive datatypes. However, working in such a general setting is notationally tedious. Moreover, all the phenomena we are interested in can already be observed over the natural numbers. Hence, we restrict ourselves in this article to a one-sorted setting over the natural numbers. By $0/0$ and $s/1$ we denote function symbols that represent the number zero and the successor function on natural numbers, respectively. We fix some abbreviations. Let n be a natural number and t a term, then $s^n(t)$ denotes the term $\underbrace{s(\cdots s(t)\cdots)}_{n \text{ times}}$ and $\overline{n}$ denotes the term $s^n(0)$. Furthermore, let $+$ be a binary infix function symbol representing addition of natural numbers, then the notation $n \cdot t$ for the multiplication of the term t by the constant n is defined inductively by $0 \cdot t = 0$ and $(i+1) \cdot t = t + (i \cdot t)$.

Definition 7. Let $\varphi(x, \vec{z})$ be a formula, then $I_x\varphi$ denotes the formula

$$\varphi(0, \vec{z}) \wedge (\forall x)(\varphi(x, \vec{z}) \to \varphi(s(x), \vec{z})) \to (\forall x)\varphi(x, \vec{z}).$$

In the definition above, we call φ the induction formula, x the induction variable, and $\vec{z}$ the induction parameters. Let Γ be a set of formulas, then the theory Γ-IND is axiomatized by the universal closure of the formulas $I_x\gamma$ with $\gamma \in \Gamma$.

If induction is carried out on formulas without induction parameters, we speak of parameter-free induction. A notion related to the induction scheme is that of inductivity in a theory.

Definition 8. Let T be a theory. A formula $\varphi(x, \vec{z})$ is T-inductive in x if $T \vdash \varphi(0, \vec{z})$ and $T \vdash \varphi(x, \vec{z}) \to \varphi(s(x), \vec{z})$. Whenever the induction variable x is clear from the context we simply say that φ is inductive in T.

Let us now introduce the setting of linear arithmetic. This setting has the advantage of being sufficiently complex to provide interesting independence results while still having straightforward model theoretic properties.

Definition 9 *(Language of linear arithmetic).* The function symbol $p/1$ represents the predecessor function on natural numbers and the infix function symbol $+/2$ represents the addition of natural numbers. The language L_{LA} of linear arithmetic is $\{0, s, p, +\}$.

By $\mathbb{N}$ we denote the set of natural numbers as well as the L_{LA} structure whose domain is the set of natural numbers and that interprets the symbols 0, s, $+$ naturally and interprets the symbol p by $p^{\mathbb{N}}(0) = 0$ and $p^{\mathbb{N}}(n+1) = n$ for all $n \in \mathbb{N}$. Analogously, we denote by $\mathbb{Z}$ the set of integers and the L_{LA} structure whose domain consists of the integers and that interprets all symbols naturally. In particular, $\mathbb{Z}$ interprets the symbol p as the function $x \mapsto x - 1$. All the theories of linear arithmetic that we will work with are extensions of the following base theory.

Definition 10. The L_{LA} theory B is axiomatized by the universal closure of the formulas

$$s(0) \neq 0, \tag{A1}$$
$$p(0) = 0, \tag{A2}$$
$$p(s(x)) = x, \tag{A3}$$
$$x + 0 = x, \tag{A4}$$
$$x + s(y) = s(x + y). \tag{A5}$$

In the following we will recall some basic properties of the theory B and its extension by induction for quantifier-free formulas. Clearly, we have $\mathbb{N} \models B$ and $\mathbb{Z} \models \mathrm{A1} + \mathrm{A3} + \mathrm{A4} + \mathrm{A5}$, but $\mathbb{Z} \not\models \mathrm{A2}$, since $\mathbb{Z} \models p(0) = -1$.

Lemma 11.

(i) $\mathrm{A1} + \mathrm{A2} + \mathrm{A3} \vdash s(x) \neq 0$.
(ii) $\mathrm{A3} \vdash s(x) = s(y) \to x = y$.

Proof. For *(i)* assume that there exists x such that $s(x) = 0$, then by (A2) and (A3) we have $x = p(s(x)) = p(0) = 0$. Thus, $s(0) = 0$ which contradicts (A1). For *(ii)* assume $s(x) = s(y)$, then we have $p(s(x)) = p(s(y))$ and by (A3) we obtain $x = y$. □

Definition 12. Let T be an L_{LA} theory. We say T is sound if $\mathbb{N} \models T$. Furthermore, T is $\exists_1$-complete if $\mathbb{N} \models \sigma$ implies $T \vdash \sigma$ for all $\exists_1$ L_{LA} sentences σ.

Lemma 13. *Let t be an L_{LA} ground term, then there exists $k \in \mathbb{N}$ such that $B \vdash t = \overline{k}$.*

Proof. Proceed by induction on the structure of the term. □

Lemma 14. *Let φ be a quantifier-free L_{LA} sentence, then either $B \vdash \varphi$ or $B \vdash \neg\varphi$.*

Proof. By a straightforward induction on the structure of the sentence φ. The only interesting case is the case where φ is an atom $t_1 = t_2$. By Lemma 13 there exist $k_1, k_2 \in \mathbb{N}$ such that $B \vdash t_1 = t_2 \leftrightarrow \overline{k_1} = \overline{k_2}$. If $k_1 = k_2$ we apply reflexivity. Otherwise, we apply Lemma 11.*(ii)* repeatedly and finally we use 11.*(i)*. □

Lemma 15. *The theory B is $\exists_1$-complete.*

Proof. By Lemma 14, B is also complete for quantifier-free sentences. Assume that $\mathbb{N} \models (\exists\vec{x})\varphi(\vec{x})$, where φ is quantifier-free. Then there are $n_1, \ldots, n_k$ such that $\mathbb{N} \models \varphi(\overline{n_1}, \ldots, \overline{n_k})$. Therefore, $B \vdash \varphi(\overline{n_1}, \ldots, \overline{n_k})$, thus, $B \vdash (\exists\vec{x})\varphi(\vec{x})$. □

Lemma 16. *The theory $B + \mathrm{Open}(L_{\mathrm{LA}})$-IND proves the following formulas*

$$x = 0 \vee x = s(p(x)), \tag{B1}$$
$$x + y = y + x, \tag{B2}$$
$$x + (y + z) = (x + y) + z, \tag{B3}$$
$$x + y = x + z \to y = z. \tag{B4}$$

Proof. Routine. □

Definition 17. The theory B' is axiomatized by (A1)–(A5) and (B1)–(B4).

Theorem 18 *([31]).* $B + \mathrm{Open}(L_{\mathrm{LA}})$-IND $\equiv B'$.

3. Analysis of clause set cycles

In this section we carry out an analysis of the formalism of refutation by a clause set cycle. In Section 3.1 we define clause set cycles and recall some basic properties as well as some results from [18]. After that, we will provide in Section 3.2 a characterization of clause set cycles in terms of a logical theory with induction. Finally, in Section 3.3 we will use this characterization and an independence result, that will be proved in Section 4, to obtain concrete and practically meaningful unrefutability results for clause set cycles.

3.1. Clause set cycles

Refutation by a clause set cycle is a formalism introduced in [18] by the authors of this article to describe abstractly the inductive arguments that take place in the n-clause calculus [23,22]. The n-clause calculus is an extension of the superposition calculus by a mechanism that detects cyclic dependencies between the derived clauses. These cyclic dependencies correspond to arguments by infinite descent and thus establish the inductive unsatisfiability of a set of clauses. The notion of refutation by a clause set cycle abstracts the underlying superposition calculus and the detection of the cycle in that proof system and therefore extracts the essence of the arguments by infinite descent that may appear in refutations by the n-clause calculus.

Since all the variables occurring in clauses are implicitly universally quantified, a clause set does not have a free variable on which we can carry out an argument by induction. Instead we will rely on a special constant symbol η, on which arguments by infinite descent will take place. This is in analogy to the special constant n that is used by the n-clause calculus for the same purpose, see [23]. The constant η can be thought of as a Skolem constant, that is selected before a refutation is attempted. In particular, clauses may of course contain other Skolem symbols besides η.

Carrying out arguments by infinite descent (or induction) only on positions of constants is unsurprisingly very restricting (see Corollary 49). Since clause set cycles are used as an abstraction of the inductive cycles of the n-clause calculus, we did not extend the formalism to allow arguments to take place in more varied positions. The logical characterization that we give in Section 3.2 makes considering such extensions easier. In particular, the main unprovability result of this article, Corollary 55, does not rely on this restriction. A method that lifts this restriction has been proposed in [13].

Remark 19. In the literature [23,22,18] a constant such as η is usually called a parameter. In order to avoid confusion with induction parameters in the sense of Definition 7 we will not use this designation.

Let $\mathcal{C}$ be a clause set possibly containing η, then we write $\mathcal{C}(\eta)$ to indicate all the occurrences of η in $\mathcal{C}$. Let furthermore t be a term, then $\mathcal{C}(t)$ denotes the clause set obtained by replacing all the occurrences of η in $\mathcal{C}$ by t.

Definition 20 *(Refutation by a clause set cycle).* Let L be a first-order language. A finite $L \cup \{\eta\}$ clause set $\mathcal{C}(\eta)$ is called an L clause set cycle if it satisfies the following conditions

$$\mathcal{C}(s(\eta)) \models \mathcal{C}(\eta), \tag{C1}$$

$$\mathcal{C}(0) \models \bot. \tag{C2}$$

Let $\mathcal{D}(\eta)$ be an $L \cup \{\eta\}$ clause set, then $\mathcal{D}(\eta)$ is refuted by an L clause set cycle $\mathcal{C}(\eta)$ if

$$\mathcal{D}(\eta) \models \mathcal{C}(\eta). \tag{C3}$$

A clause set cycle represents an argument by infinite descent in the following sense. Suppose there is an $L \cup \{\eta\}$ structure M with $D(M) = \mathbb{N}$ such that $M \models \mathcal{C}(\eta)$. By (C2) we have $\eta^M > 0$. Now let $m \in \mathbb{N}$, then we denote by $M[\eta \mapsto m]$ the $L \cup \{\eta\}$ structure with the same domain as M, that interprets all non-logical symbols except η as M, and interprets η as m. Then we have $M[\eta \mapsto \eta^M - 1] \models \mathcal{C}(s(\eta))$ and by (C1) we now obtain $M[\eta \mapsto \eta^M - 1] \models \mathcal{C}(\eta)$. Hence, we obtain an finite strictly descending sequence of natural numbers m such that $M[\eta \mapsto m] \models \mathcal{C}(\eta)$. This is impossible, hence $M \not\models \mathcal{C}(\eta)$.

Remark 21. In the literature cycles on clause sets are usually equipped with parameters that control the offset and the descent step size and thus permit a more flexible usage of the cycles (see for example [23]). In Definition 24 we shall consider clause set cycles with parameters inspired by the parameters found in the cycles of the n-clause calculus. After that, we show in Proposition 26 that such parameters do not make the system more powerful. In particular, [18] uses a slightly different notation. A refutation by a clause set cycle in [18] corresponds to a refutation by a $(1, 0)$-clause set cycle with external offset $i \in \mathbb{N}$ in the sense of Definition 24. Hence, by Proposition 26 the notion of refutation by clause set cycle used in [18] is exactly as powerful as the more elegant notion of refutation by a clause set cycle used in this article.

Clause set cycles could be integrated into a saturation-based prover by carrying out the saturation process as usual and by detecting a clause set cycles among the clauses derived so far, thus satisfying Condition (C3) with respect to the set of

generated clauses. The detection of a clause set cycle could for example make use of the derivation relation generated by the prover in order to detect the Conditions (C1) and (C2). The detection of a clause set cycle, then provides the inductive unsatisfiability of the clauses generated and therefore ends the refutation. This is essentially how the n-clause calculus described in [23,22] operates.

Let us now consider an example of a refutation by a clause set cycle.

Definition 22. By $\mathsf{C}(\eta)$ we denote the $L_{\mathrm{LA}} \cup \{\eta\}$ clause set

$$cls(B + \mathrm{B2}) \cup \{\{\eta \neq x + x\}, \{\eta \neq s(x + x)\}\}.$$

Example 23. Intuitively, the clause set $\mathsf{C}(\eta)$ asserts the existence of an element η, which is neither even nor odd. We will now show that $\mathsf{C}(\eta)$ is a clause set cycle.

We start by showing that $\mathsf{C}(\eta)$ satisfies Condition (C2). Suppose that $\mathsf{C}(0)$ has a model M, then we have in particular $M \models 0 \neq 0 + 0 = 0$. This is a contradiction, and therefore $\mathsf{C}(0) \models \bot$.

For Condition (C1), let M be a model of $\mathsf{C}(s(\eta))$. Clearly, we have $M \models cls(B + \mathrm{B2})$, hence we only have to show that $M \models \eta \neq x + x$ and $M \models \eta \neq s(x + x)$. Suppose that $M \models \eta = d + d$ for some $d \in M$, then we have $M \models s(\eta) = s(d + d)$. Since $M \models \mathsf{C}(s(\eta))$, we also have $M \models s(\eta) \neq s(d + d)$, a contradiction. Now suppose that $M \models \eta = s(d + d)$ for some $d \in M$. Since $M \models \mathsf{C}(s(\eta))$, we also have $M \models s(\eta) = s(s(d + d)) = s(d) + s(d)$. Thus $M \models \mathsf{C}(\eta)$, that is, $\mathsf{C}(s(\eta)) \models \mathsf{C}(\eta)$.

Hence, $\mathsf{C}(\eta)$ is a clause set cycle and therefore refutes itself.

The induction argument contained in a refutation by a clause set cycle is peculiar in the sense that it does not take place in an explicit background theory. Instead of a background theory clause set cycles may contain clauses free of η that act as a background theory. In the example above the clause set cycle contains the clauses $cls(B + \mathrm{B2})$, that correspond to the background theory.

The cycles detected by practical methods such as the n-clause calculus differ from clause set cycles in that they can be controlled by three parameters: An external offset, an internal offset, and the step size of the descent. In the following we will show that these parameters do not increase the overall strength of the system.

Definition 24. Let L be a first-order language and $j, k \in \mathbb{N}$ with $j \geq 1$. A finite $L \cup \{\eta\}$ clause set $\mathcal{C}(\eta)$ is called an L (j,k)-clause set cycle if

$$\mathcal{C}(s^{j+k}(\eta)) \models \mathcal{C}(s^k(\eta)), \tag{C1'}$$

$$\mathcal{C}(\overline{m+k}) \models \bot, \text{ for } m = 0, \ldots, j-1. \tag{C2'}$$

We call the parameter j the descent step size and k the internal offset. Let $i \in \mathbb{N}$ and $\mathcal{D}(\eta)$ an $L \cup \{\eta\}$ clause set, then $\mathcal{D}(\eta)$ is refuted by the (j,k)-clause set cycle $\mathcal{C}(\eta)$ with external offset i, if

$$\mathcal{D}(s^i(\eta)) \models \mathcal{C}(s^k(\eta)), \tag{C3'}$$

$$\mathcal{D}(\overline{m}) \models \bot, \text{ for } m = 0, \ldots, i-1. \tag{C3"}$$

Clearly, clause set cycles in the sense of Definition 20 are exactly the $(1,0)$-clause set cycles and a refutation by a clause set cycle in the sense of Definition 20 is a refutation by a $(1,0)$-clause set cycle with external offset 0.

We start by showing that (j,k)-clause set cycles with $j, k \in \mathbb{N}$ and $j \geq 1$ can be simulated by clause set cycles.

Lemma 25. *Let L be a first-order language, $j, k \in \mathbb{N}$ with $j \geq 1$, and $\mathcal{C}(\eta)$ an L (j,k)-clause set cycle. Then there exists a clause set cycle $\mathcal{C}'(\eta)$ such that $\mathcal{C}(s^k(\eta)) \models \mathcal{C}'(\eta)$.*

Proof. We start by eliminating the internal offset of the (j,k)-clause set cycle, by letting $\mathcal{C}'(\eta) := \mathcal{C}(s^k(\eta))$. It is clear that $\mathcal{C}'$ is a $(j,0)$-clause set cycle. Moreover by the definition of $\mathcal{C}'$ we have $\mathcal{C}(s^k(\eta)) \models \mathcal{C}'(\eta)$. Let $\mathcal{C}''(\eta)$ be the clause set obtained by applying Lemma 6 to the set $\mathfrak{C} := \{\mathcal{C}'(s^m(\eta)) \mid m = 0, \ldots, j-1\}$. We will now show that $\mathcal{C}''(\eta)$ is a clause set cycle. Suppose that $M \models \mathcal{C}''(0)$, then $M \models \mathcal{C}'(\overline{m})$ for some $m \in \{0, \ldots, j-1\}$, which is impossible and therefore $\mathcal{C}''(0) \models \bot$. Now suppose that $M \models \mathcal{C}''(s(\eta))$. Then we have $M \models \mathcal{C}'(s^{m+1}(\eta))$ for some $m \in \{0, \ldots, j-1\}$. If $m+1 \leq j-1$, then $\mathcal{C}(s^{m+1}(\eta)) \in \mathfrak{C}$ and therefore $M \models \mathcal{C}''(\eta)$. Otherwise we have $m+1 = j$ and therefore by (C1') we obtain $M \models \mathcal{C}'(\eta)$ and since $\mathcal{C}'(\eta) \in \mathfrak{C}$, we have $\mathcal{C}'(\eta) \models \mathcal{C}''(\eta)$. □

Now we can show that a refutation by a (j,k)-clause set cycle with internal offset i, where $i, j, k \in \mathbb{N}$ with $j \geq 1$ can be reduced to a refutation by a clause set cycle.

Proposition 26. *Let L be a first-order language, $\mathcal{D}(\eta)$ an $L \cup \{\eta\}$ clause set, and $i, j, k \in \mathbb{N}$ with $j \geq 1$ such that $\mathcal{D}$ is refuted by an L (j,k)-clause set cycle with external offset i. Then $\mathcal{D}(\eta)$ is refuted by a clause set cycle.*

Proof. Let $\mathcal{C}(\eta)$ be an L (j,k)-clause set cycle such that $\mathcal{D}(\eta)$ is refuted by $\mathcal{C}$ with external offset i. By Lemma 25 there exists a clause set cycle $\mathcal{C}'(\eta)$ such that $\mathcal{C}(s^k(\eta)) \models \mathcal{C}'(\eta)$. Hence $\mathcal{D}$ is refuted by a $(1,0)$-clause set cycle with external offset i. In the next step we will eliminate the external offset. Let $\mathfrak{C} := \{\mathcal{D}(s^m(\eta)) \mid m = 0,\dots,i-1\} \cup \{\mathcal{C}'(\eta)\}$ and apply Lemma 6 in order to obtain a clause set $\mathcal{C}''(\eta)$ corresponding to the disjunction of the clause sets in $\mathfrak{C}$. We will now show that $\mathcal{C}''(\eta)$ is a clause set cycle. Suppose that $M \models \mathcal{C}''(0)$, then either $M \models \mathcal{D}(s^m(\eta))$ for some $m \in \{0,\dots,i-1\}$ or $M \models \mathcal{C}'(0)$. The first case is impossible because of Condition (C3") and the second case is impossible because $\mathcal{C}'(\eta)$ is a clause set cycle and therefore $\mathcal{C}'(0) \models \bot$. Hence we have $\mathcal{C}''(0) \models \bot$. Now suppose that $M \models \mathcal{C}''(s(\eta))$. If $M \models \mathcal{C}'(s(\eta))$, then we have $M \models \mathcal{C}'(\eta)$ because $\mathcal{C}'$ is a clause set cycle and therefore $M \models \mathcal{C}''(\eta)$. If $M \models \mathcal{D}(s^{m+1}(\eta))$ for some $m \in \{0,\dots,i-1\}$, we need to consider two cases. If $m+1 < i$, then we have $\mathcal{D}(s^{m+1}(\eta)) \in \mathfrak{C}$ and therefore $M \models \mathcal{C}''(\eta)$. Otherwise we have $m+1 = i$, and therefore we obtain $M \models \mathcal{C}'(\eta)$ by Condition ((C3')). Again we obtain $M \models \mathcal{C}''(\eta)$. Hence $\mathcal{C}''(\eta)$ is a clause set cycle. We complete the proof by observing that $\mathcal{D}(\eta) \models \mathcal{C}''(\eta)$, since $\mathcal{D}(\eta) \in \mathfrak{C}$. Hence, $\mathcal{D}(\eta)$ is refuted by the clause set cycle $\mathcal{C}''(\eta)$. □

As already mentioned earlier, the notion of refutation by a clause set cycle is a useful intermediary abstraction of the induction mechanism of a family of AITP systems including in particular the n-clause calculus [23,22]. Since our goal is to develop a uniform logical representation of methods for AITP, we thus use the notion of refutation by a clause set cycle as a starting point to provide logical abstractions of AITP systems such as the n-clause calculus. In particular, we want, for a fixed language L, to provide a logical $L \cup \{\eta\}$ theory T that simulates refutation by a clause set cycle in the following sense: Let $\mathcal{D}(\eta)$ be an $L \cup \{\eta\}$ clause set that is refuted by an L clause set cycle, then $T + \mathcal{D}(\eta)$ is inconsistent. The authors of this article have shown in [18] that refutation by L clause set cycles can be simulated by the theory $\exists_1(L)$-IND (see Theorem 42) and moreover that Open(L)-IND does not simulate refutations by a clause set cycle.

Theorem 27 *([18, Theorem 4.6]). There exists a language L and an $L \cup \{\eta\}$ clause set $\mathcal{D}(\eta)$ such that $\mathcal{D}(\eta)$ is refuted by an L clause set cycle, but* Open(L)-IND $+ \mathcal{D}(\eta)$ *is consistent.*

In the following section, we will give a proof of Theorem 27 that is simpler, shorter, and more elegant than the proof given in [18].

Definition 28. Let n, m be natural numbers, then by $n \dot{-} m$ we denote the truncated subtraction of m from n given by

$$n \dot{-} m := \begin{cases} n - m & \text{if } n \geq m \\ 0 & \text{otherwise} \end{cases}.$$

Lemma 29. $B +$ Open(L_{LA})-IND $\nvdash (\exists y)(x = y + y \vee x = s(y + y))$

Proof. By Theorem 18 it suffices to show that $B' \nvdash (\exists y)(x = y + y \vee x = s(y + y))$. Consider the L_{LA} structure M whose domain consists of the pairs of the form $(m,n) \in \mathbb{N} \times \mathbb{Z}$ such that $m = 0$ implies $n \in \mathbb{N}$ and that interprets the non-logical symbols as follows:

$$0^M = (0,0),$$
$$s^M((m,n)) = (m, n+1),$$
$$p^M((m,n)) = \begin{cases} (m, n \dot{-} 1) & \text{if } m = 0, \\ (m, n-1) & \text{otherwise} \end{cases},$$
$$(m_1,n_1) +^M (m_2,n_2) = (m_1 + m_2, n_1 + n_2).$$

It is routine to verify that $M \models B'$. Consider the element $(1,0)$, then clearly there is no element (m,n) of M such that $(1,0) = (m,n) +^M (m,n) = (2m,2n)$ or $(1,0) = s^M((m,n) +^M (m,n)) = (2m, 2n+1)$. □

Proof of Theorem 27. Consider the clause set $\mathsf{C}(\eta)$. In Example 23 we have shown that $\mathsf{C}(\eta)$ is refuted by an L_{LA} clause set cycle. We will now show that Open(L_{LA})-IND $+ \mathsf{C}(\eta)$ is consistent. We proceed indirectly and assume that Open(L_{LA})-IND $+$ $\mathsf{C}(\eta)$ is inconsistent. Hence $B +$ B2 $+$ Open(L_{LA})-IND $\vdash (\exists y)(\eta = y + y) \vee (\exists y)(\eta = s(y + y))$. Thus, $B +$ Open(L_{LA})-IND $\vdash$ $(\exists y)(x = y + y \vee x = s(y + y))$, which contradicts Lemma 29. □

However, empirical evidence suggests that clause set cycles are not strictly stronger than open induction. This has given rise to the following conjecture.

Conjecture 30 *([18, Conjecture 4.7]). There exists a language L and an $L \cup \{\eta\}$ clause set $\mathcal{D}(\eta)$ such that* Open(L)-IND $\cup \mathcal{D}(\eta)$ *is inconsistent, but $\mathcal{D}(\eta)$ is not refuted by an L clause set cycle.*

In the following section we will give a characterization of refutation by a clause set cycle in terms of a logical theory. In Section 3.3 we will make use of this characterization to give a positive answer to Conjecture 30.

3.2. Logical characterization

In the previous section we have introduced the notion of refutation by a clause set cycle and we have shown that certain practically motivated generalizations of refutation by a clause set cycle do not result in stronger systems. In this section we will give a characterization of refutation by a clause set cycle in terms of a logical theory.

We start by converting clause set cycles into formulas.

Lemma 31. *Let $\mathcal{C}(\eta)$ be an L clause set cycle, then the formula $\neg cls^{-1}(\mathcal{C})[\eta/x]$ is $\varnothing$-inductive. Let $\mathcal{D}(\eta)$ be an $L \cup \{\eta\}$ clause set that is refuted by the clause set cycle $\mathcal{C}(\eta)$, then $\neg cls^{-1}(\mathcal{C}) + \mathcal{D}(\eta)$ is inconsistent.*

Proof. Clearly, we have $M \models \neg cls^{-1}(\mathcal{C})$ if and only if $M \not\models \mathcal{C}$. Hence, $\models \neg cls^{-1}(\mathcal{C}(0))$ and $\neg cls^{-1}(\mathcal{C}(\eta)) \models \neg cls^{-1}(\mathcal{C}(s(\eta)))$. Therefore, by the completeness theorem and the deduction theorem for first-order logic we have

$$\vdash \neg cls^{-1}(\mathcal{C}(0)),$$
$$\vdash \neg cls^{-1}(\mathcal{C}(\eta)) \to \neg cls^{-1}(\mathcal{C}(s(\eta))).$$

Thus, $\vdash \neg cls^{-1}(\mathcal{C})[\eta/0]$ and $\vdash \neg cls^{-1}(\mathcal{C})[\eta/x] \to \neg cls^{-1}(\mathcal{C})[\eta/s(x)]$. The second part of the lemma is obvious. □

Let $\mathcal{C}$ be a clause set cycle, then the formula $\neg cls^{-1}(\mathcal{C})[\eta/x]$ is the formula that corresponds to the induction argument contained in a refutation by a clause set cycle. Clearly, this formula is logically equivalent to an $\exists_1$ formula. In the following we will make three further important observations about this argument by induction.

The first observation is that the formula $\neg cls^{-1}(\mathcal{C})[\eta/x]$ has only one free variable, that is, the variable on which the argument by induction takes places. Hence the induction captured by clause set cycles is essentially parameter-free induction. In this article we use a notation for parameter-free induction that is inspired by the notation used in the literature from mathematical logic on parameter-free induction [1,24,4,5,8].

Definition 32. Let Γ be a set of formulas, then Γ-IND$^-$ is axiomatized by the universal closure of the formulas $I_x\varphi$ for $\varphi(x) \in \Gamma$.

When the set of induction formulas is unrestricted, induction without parameters is just as powerful as induction with parameters.

Lemma 33. *Let L be a first-order language, then we have*

$$\mathcal{F}(L)\text{-IND} \equiv \mathcal{F}(L)\text{-IND}^-.$$

Proof. We only show $\mathcal{F}(L)$-IND$^- \vdash \mathcal{F}(L)$-IND, the other direction is trivial. Let $\varphi(x, \vec{z})$ be an L formula, x a variable, and $\vec{z}$ a vector of variables. We let the formula $\psi(x)$ be given by

$$(\forall \vec{z})(\varphi(0, \vec{z}) \wedge (\forall x)(\varphi(x, \vec{z}) \to \varphi(s(x), \vec{z})) \to \varphi(x, \vec{z})).$$

By a straightforward quantifier shift we obtain $\vdash (\forall x)\psi(x) \leftrightarrow (\forall \vec{z}) I_x \varphi(x, \vec{z})$. Furthermore, it is straightforward to check that $\vdash \psi(0)$ and $\vdash \psi(x) \to \psi(s(x))$. Hence, $\vdash I_x\psi \to (\forall x)\psi$. Therefore $\vdash I_x\psi \to (\forall \vec{z}) I_x \varphi$. □

However, when we are dealing with restricted induction schemes such as $\exists_k(L)$-IND, then its parameter-free counterpart $\exists_k(L)$-IND$^-$ may be a weaker theory [24].

Another remarkable property of the formula $\neg cls^{-1}(\mathcal{C})[\eta/x]$ is its $\varnothing$-inductivity. In a refutation by a clause set cycle, there is no explicit induction axiom. Instead, whenever a clause set $\mathcal{C}(\eta)$ is shown to be a clause set cycle, it can be used in a refutation. This is reminiscent of a Hilbert-style induction rule that allows us to deduce $cls^{-1}(\mathcal{C})[\eta/x]$ if $cls^{-1}(\mathcal{C})[\eta/x]$ is $\varnothing$-inductive. The idea of Hilbert-style inference rules and in particular of induction rules is made explicit in the following two definitions.

Definition 34. An inference rule R is a set of tuples of the form Γ/γ_0 called the instances of R, where $\Gamma = \{\gamma_1, \ldots, \gamma_n\}$ is a finite set of sentences and γ_0 is a sentence. Let T be a theory, then the theory of unnested applications $[T, R]$ of the inference rule R over the theory T is axiomatized by

$$T + \{\varphi \mid T \vdash \Gamma, \Gamma/\varphi \in R\}.$$

Let $[T, R]_0 := T$ and $[T, R]_{n+1} = [[T, R]_n, R]$, then we define $T + R := \bigcup_{n \geq 0} [T, R]_n$.

Let R be an inference rule and $\Gamma/\gamma_0 \in R$, then the intended meaning of the rule instance Γ/γ_0 is that whenever all the sentences in Γ are derived, then we can derive γ_0. The instance Γ/γ_0 will also be written as

$$\frac{\gamma_1 \quad \cdots \quad \gamma_n}{\gamma_0}$$

Definition 35. Let Γ be a set of formulas, then the rule Γ-IND^R consists of the instances of the form

$$\frac{(\forall \vec{z})\gamma(0,\vec{z}) \qquad (\forall \vec{z})(\forall x)(\gamma(x,\vec{z}) \to \gamma(s(x),\vec{z}))}{(\forall \vec{z})(\forall x)\gamma(x,\vec{z})},$$

with $\gamma \in \Gamma$ and where the variable x is called the induction variable and the variables $\vec{z}$ are called the induction parameters. The induction rule Γ-IND^{R-} consists of these instances of Γ-IND^R where the induction variable is the only free variable of the induction formula.

Let T be a theory and Γ a set of formulas, then we can make use of Definition 8 to reformulate the theory $[T, \Gamma\text{-IND}^R]$ as follows

$$[T, \Gamma\text{-IND}^R] \equiv T + \{\varphi \mid \varphi(x,\vec{z}) \in \Gamma, \varphi \text{ is } T\text{-inductive in } x\}.$$

In other words the theory $[T, \Gamma\text{-IND}^R]$ provides induction only for T-inductive formulas from Γ, whereas $T + \Gamma$-IND provides induction for all formulas in Γ. It is obvious that $T + \Gamma\text{-IND} \vdash [T, \Gamma\text{-IND}^R]$. However, $[T, \Gamma\text{-IND}^R]$ is in general not as strong as $T + \Gamma$-IND, see [26]. For further literature on induction rules, see for example [31,30,26,3,20].

We will now make a last observation about the argument by induction contained in a refutation by a clause set cycle. The previous observations show that clause set cycles are simulated by unnested applications of the parameter-free $\exists_1$ induction rule over the theory $\varnothing$. A sentence derived by an induction rule is the universal closure of an inductive formula. Hence, once a formula is derived by an induction rule it can be instantiated freely. Similarly, a clause set cycle $\mathcal{C}(\eta)$ acts, roughly speaking, as the lemma $\neg cls^{-1}(\mathcal{C})[\eta/x]$ of which, however, only the instance $\neg cls^{-1}(\mathcal{C})$ is used. In other words, a clause set cycle allows us to derive properties of η only. We will informally refer to this restriction as the instance restriction. We can capture this restriction in the following restricted induction rule.

Definition 36. Let Γ be a set of formulas, then the rule Γ-IND$^R_\eta$ consists of the instances of the form

$$\frac{(\forall \vec{z})\gamma(0,\vec{z}) \qquad (\forall \vec{z})(\forall x)(\gamma(x,\vec{z}) \to \gamma(s(x),\vec{z}))}{(\forall \vec{z})\gamma(\eta,\vec{z})}, \text{ with } \gamma \in \Gamma.$$

The rule Γ-IND$^{R-}_\eta$ consists of those instances of Γ-IND$^R_\eta$ where the induction variable is the only free variable of the induction formula.

By combining the above observations we obtain the following proposition, that allows us to simulate clause set cycles in a logical theory.

Proposition 37. *Let $\mathcal{D}(\eta)$ be an $L \cup \{\eta\}$ clause set. If $\mathcal{D}(\eta)$ is refuted by an L clause set cycle, then $[\varnothing, \exists_1(L)\text{-IND}^{R-}_\eta] \cup \mathcal{D}(\eta)$ is inconsistent.*

Proof. Since $\mathcal{D}$ is refuted by a clause set cycle, there exists an L clause set cycle $\mathcal{C}(\eta)$ such that

$$\mathcal{D}(\eta) \models \mathcal{C}(\eta). \tag{*}$$

Let $\varphi(x) \coloneqq cls^{-1}(\mathcal{D})[\eta/x]$ then $\varphi(\eta)$ is clearly logically equivalent to $\mathcal{D}(\eta)$. By the soundness of first-order logic it thus suffices to show that

$$[\varnothing, \exists_1(L)\text{-IND}^{R-}] \vdash \neg\varphi(\eta).$$

Let $\psi(x)$ be an $\exists_1$ formula that is logically equivalent to $\neg cls^{-1}(\mathcal{C})[\eta/x]$. Then, by applying the completeness theorem and the deduction theorem to (*), we obtain

$$\vdash \varphi(\eta) \to \neg\psi(\eta). \tag{†}$$

By Lemma 31 we know that $\psi(x)$ is $\varnothing$-inductive, and therefore we have $[\varnothing, \exists_1(L)\text{-IND}^{R-}_\eta] \vdash \psi(\eta)$. Hence, by considering the contrapositive of (†) we clearly obtain $[\varnothing, \exists_1(L)\text{-IND}^{R-}_\eta] \vdash \neg\varphi(\eta)$. □

We will now show that we even have the converse and thus obtain a characterization of refutation by a clause set cycle by a logical theory. We start by observing that finitely many inductive formulas can be fused into a single inductive formula.

Lemma 38. *Let T be a theory and let $\varphi_1(x,\vec{z}),\dots,\varphi_n(x,\vec{z})$ be formulas. If φ_i is T-inductive in x for $i=1,\dots,n$, then $\psi \coloneqq \bigwedge_{i=1,\dots n}\varphi_i$ is T-inductive in x.*

Proof. We start by showing that $T \vdash \psi(0,\vec{z})$. Let $j\in\{1,\dots,n\}$, then since φ_j is T-inductive in x, we have $T\vdash\varphi_j(0,\vec{z})$ and we are done. Now let us show that $T\vdash\psi(x,\vec{z})\to\psi(s(x),\vec{z})$. Work in T, assume $\bigwedge_{i=1}^n\varphi_i(x,\vec{z})$, and let $j\in\{1,\dots,n\}$. Since φ_j is T-inductive in x, we have $\varphi_j(x,\vec{z})\to\varphi_j(s(x),\vec{z})$. Hence we obtain $\varphi_j(s(x),\vec{z})$ and therefore ψ is T-inductive in x. □

This simple result is particularly interesting because fusing inductive formulas neither introduces more induction parameters and when fusing $\exists_k$ induction formulas, the fused induction formula is also logically equivalent to an $\exists_k$ formula. Similar techniques exist for fusing a finite number of induction axioms into a single induction axiom [19,14]. However, these either introduce a new induction parameter or increase the quantifier complexity of the resulting induction formula.

Proposition 39. *Let $\mathcal{D}(\eta)$ be an $L\cup\{\eta\}$ clause set. If $[\varnothing,\exists_1(L)\text{-IND}_\eta^{R-}]+\mathcal{D}(\eta)$ is inconsistent, then $\mathcal{D}(\eta)$ is refuted by an L clause set cycle.*

Proof. Let $\varphi(x)\coloneqq cls^{-1}(\mathcal{D})[\eta/x]$, then by the completeness theorem and the deduction theorem we obtain $[\varnothing,\exists_1(L)\text{-IND}_\eta^{R-}]\vdash\neg\varphi(\eta)$. By the compactness theorem there exist $\exists_1$ L formulas $\psi_1(x)$, …, $\psi_k(x)$ such that ψ_i is $\varnothing$-inductive for $i=1,\dots,k$ and

$$\psi_1(\eta)+\dots+\psi_k(\eta)\vdash\neg\varphi(\eta).$$

By Lemma 38, the formula $\Psi(x)\coloneqq\bigwedge_{i=1}^k\psi_i$ is $\varnothing$-inductive. Moreover we have $\Psi(\eta)\vdash\neg\varphi(\eta)$. Clearly, Ψ is logically equivalent to an $\exists_1$ formula, hence there exists a $\forall_1$ formula Θ that is logically equivalent to $\neg\Psi$. Since $\vdash\Psi(0)$ and $\vdash\Psi(x)\to\Psi(s(x))$, we have $\Theta(0)\models\bot$ and $\Theta(s(x))\models\Theta(x)$. Therefore, $\mathcal{C}\coloneqq cls(\Theta(\eta))$ is a clause set cycle. Finally, since $\Psi(\eta)\vdash\neg\varphi(\eta)$, we obtain $\varphi(\eta)\models\neg\Psi(\eta)$, that is, $\mathcal{D}(\eta)\models\mathcal{C}(\eta)$. In other words, $\mathcal{D}$ is refuted by the clause set cycle $\mathcal{C}$. □

We thus obtain a characterization of refutation by a clause set cycle in terms of induction rules.

Theorem 40. *Let $\mathcal{D}(\eta)$ be an $L\cup\{\eta\}$ clause set, then $\mathcal{D}$ is refuted by an L clause set cycle if and only if $[\varnothing,\exists_1(L)\text{-IND}_\eta^{R-}]+\mathcal{D}(\eta)$ is inconsistent.*

Proof. An immediate consequence of Propositions 37 and 39. □

Remark 41. In a refutation by a clause set cycle the constant η plays essentially two roles: On the one hand, it can be thought of as a Skolem symbol and, on the other hand, it plays the role of an induction variable. The characterization of Theorem 40 clarifies this situation by allowing us to distinguish between induction variables and the Skolem symbol η.

As a corollary we obtain Theorem 2.10 of [18].

Theorem 42 *([18, Theorem 2.10]). Let L be a first-order language and $\mathcal{D}(\eta)$ an $L\cup\{\eta\}$ clause set. If $\mathcal{D}(\eta)$ is refuted by an L clause set cycle, then $\exists_1(L)\text{-IND}+\mathcal{D}(\eta)$ is inconsistent.*

Proof. Obvious, since $\exists_1(L)\text{-IND}\vdash[\varnothing,\exists_1(L)\text{-IND}_\eta^{R-}]$. □

In the following section we will make use of the characterization of Theorem 40 to construct clause sets that are refutable by open induction but which are not refutable by clause set cycles. In particular the unrefutability results that we provide exploit different logical features of clause set cycles.

3.3. Unprovability by clause set cycles

In the previous sections we have introduced the notion of refutation by a clause set cycle for which we have shown a characterization in terms of a logical theory. We have shown this characterization by discerning four main logical features of refutation by a clause set cycle: the quantifier-complexity, the absence of induction parameters, the similarity with induction rules, and the restriction on instances of derived formulas. In this section we will make use of this characterization in order to provide practically relevant clause sets that are not refutable by clause set cycles, but that are refutable by induction on quantifier-free formulas. The unrefutability results in this section will exploit different logical features of clause set cycles. In particular we will show that restricting the instances of the conclusion of the induction rule can be very drastic.

Let us now briefly discuss the practical applicability of the unprovability results given in this section. The unprovability results apply to any sound (for first-order logic) saturation prover that detects clause set cycles over the language of the initial clause set. Hence, our unprovability results apply, in particular, to all sound saturation provers that do not extend the language of the initial clause set and detect cycles among the derived clauses such as for example the n-clause calculus (see [23,22]). On the other hand systems that extend the language are also of practical importance, since such extensions can be used to organize the refutation process, see for example [34]. In particular, the extension of the language by definitions can be expected to have interesting effects. However, investigating the interaction between clause set cycles and various language extending mechanisms would go beyond the scope of this article and should be investigated separately. Observe, furthermore, that our setting does not rule out the presence of Skolem symbols other than η in clause set cycles.

We start by slightly reformulating Theorem 40 so that we can work with formulas and theories instead of clause sets.

Corollary 43. *Let L be a first-order language, T a $\forall_1$ L theory, and $\varphi(x, \vec{y})$ a quantifier-free L formula, then $T + [\varnothing, \exists_1(L)\text{-IND}^{R-}_{\eta}] \vdash (\exists \vec{y})\varphi(\eta, \vec{y})$ if and only if $cls(T + (\forall \vec{y})\neg\varphi(\eta, \vec{y}))$ is refuted by an L clause set cycle.*

Proof. Clearly, $T + [\varnothing, \exists_1(L)\text{-IND}^{R-}_{\eta}] \vdash (\exists \vec{y})\varphi(\eta, \vec{y})$ if and only if $[\varnothing, \exists_1(L)\text{-IND}^{R-}_{\eta}] + cls(T + (\forall \vec{y})\neg\varphi(\eta, \vec{y}))$ is inconsistent. By Theorem 40, $[\varnothing, \exists_1(L)\text{-IND}^{R-}_{\eta}] + cls(T + (\forall \vec{y})\neg\varphi(\eta, \vec{y}))$ is inconsistent if and only if $cls(T + (\forall \vec{y})\neg\varphi(\eta, \vec{y}))$ is refuted by an L clause set cycle. □

In Section 3.1 we have informally observed that clause set cycles do not take place in some explicit background theory but instead clause set cycles contain the clauses corresponding to the background theory. In the following we will make this informal observation more precise.

Lemma 44. *Let L be a first-order language, T a $\forall_1$ L theory, U an L theory, then*

$$T + [U, \exists_1(L)\text{-IND}^{R-}_{\eta}] \equiv [T + U, \exists_1(L)\text{-IND}^{R-}_{\eta}].$$

Furthermore, $T + [U, \exists_1(L)\text{-IND}^{R-}] \equiv [T + U, \exists_1(L)\text{-IND}^{R-}]$.

Proof. The direction $[T + U, \exists_1(L)\text{-IND}^{R-}_{\eta}] \vdash T + [U, \exists_1(L)\text{-IND}^{R-}_{\eta}]$ is immediate. For the other direction let $\gamma(x)$ be an $\exists_1$ L formula and assume that $T + U \vdash \gamma(0)$ and $T + U \vdash \gamma(x) \to \gamma(s(x))$. By the compactness theorem and the deduction theorem there exist $\tau, \tau_1, \ldots, \tau_n \in T$ such that $\tau = \bigwedge_{i=1}^{n} \tau_i$ and $U \vdash \tau \to \gamma(0)$ and $U \vdash \tau \to \gamma(x) \to \gamma(s(x))$. By straightforward propositional equivalences we obtain

$$U \vdash (\tau \to \gamma(x)) \to (\tau \to \gamma(s(x))).$$

Clearly, τ is logically equivalent to a $\forall_1$ sentence, hence $\tau \to \gamma(x)$ is logically equivalent to an $\exists_1$ formula $\gamma'(x)$. Hence, $[U, \exists_1(L)\text{-IND}^{R-}_{\eta}] \vdash \gamma'(\eta)$ and therefore $[U, \exists_1(L)\text{-IND}^{R-}_{\eta}] \vdash \tau \to \gamma(\eta)$. Thus, $T + [U, \exists_1(L)\text{-IND}^{R-}_{\eta}] \vdash \gamma(\eta)$. We show $T + [U, \exists_1(L)\text{-IND}^{R-}] \equiv [T + U, \exists_1(L)\text{-IND}^{R-}]$ analogously, with the exception that in the last part of the argument we have to shift the universal quantifier in $(\forall x)(\tau \to \gamma(x))$ inwards. □

Lemma 44 allows us to move $\forall_1$ axioms in and out of the induction rule and thus to consider the η-free clauses of a clause set cycle as the background theory. As an immediate consequence of Corollary 43 and Lemma 44 we now obtain a general pattern to reduce unrefutability problems for clause set cycles to independence problems.

Proposition 45. *Let L be a first-order language, T a $\forall_1$ L theory, and let $\varphi(x, \vec{y})$ be a quantifier-free L formula. Then $[T, \exists_1(L)\text{-IND}^{R-}_{\eta}] \vdash (\exists \vec{y})\varphi(\eta, \vec{y})$ if and only if the clause set $cls(T + (\forall \vec{y})\neg\varphi(\eta, \vec{y}))$ is refuted by an L clause set cycle.*

Proof. We have the following chain of equivalences:

$$\begin{aligned}
&[T, \exists_1(L)\text{-IND}^{R-}_{\eta}] \vdash (\exists \vec{y})\varphi(\eta, \vec{y}),\\
&\quad \Leftrightarrow_{\text{Lem. } 44} T + [\varnothing, \exists_1(L)\text{-IND}^{R-}_{\eta}] \vdash (\exists \vec{y})\varphi(\eta, \vec{y})\\
&\quad \Leftrightarrow [\varnothing, \exists_1(L)\text{-IND}^{R-}_{\eta}] \cup cls(T + (\forall \vec{y})\neg\varphi(\eta, \vec{y})) \text{ is inconsistent}\\
&\quad \Leftrightarrow_{\text{Cor. } 43} cls(T + (\forall \vec{y})\neg\varphi(\eta, \vec{y})) \text{ is refuted by an } L \text{ clause set cycle.}
\end{aligned}$$ □

We can now consider some theories and formulas that will yield clause sets that are unrefutable by clause set cycles. By the characterization of clause set cycles by a logical theory we have discerned several restrictions of the induction principle that corresponds to clause set cycles. In the following two subsections we will formulate unprovability results that attack different restrictions of the induction principle that is contained in the notion of refutation by a clause set cycle.

3.3.1. Instance restriction

In Section 3.2 we have observed that a refutation by a clause set cycle only permits a single instance of a clause set cycle to appear in a refutation. In this section we will formulate an unprovability result for clause set cycles that exploits this restriction. In particular, we will base this unprovability result on a stronger independence result that shows how drastic the instance restriction is.

Definition 46. Let $f/1$ be a function symbol and $P/1$ be a predicate symbol. The theory $\mathcal{P}$ is axiomatized by the universal closure of the following formulas

$$0 \neq s(x),$$
$$s(x) = s(y) \to x = y,$$
$$P(0),$$
$$P(x) \to P(s(x)).$$

Definition 47. Let $\varphi(x, \vec{z})$ be a formula, then $I^{\eta}_{x}\varphi$ denotes the formula

$$\varphi(0, \vec{z}) \wedge (\forall x)(\varphi(x, \vec{z}) \to \varphi(s(x), \vec{z})) \to \varphi(\eta, \vec{z}).$$

Let Γ be a set of formulas, then the theory Γ-IND$_\eta$ is axiomatized by the universal closure of the formulas $I^{\eta}_{x}\gamma$ with $\gamma \in \Gamma$.

We have the following independence.

Proposition 48. $\mathcal{P} + \mathcal{F}(\{0, s, P, f\})$-IND$_\eta \nvdash P(f(\eta))$.

Proof. Let M be the $\{0, s, P, f\}$ structure with domain consisting of pairs $(m, n) \in \{0, 1\} \times \mathbb{Z}$ such that if $m = 0$, then $n \in \mathbb{N}$. Let M interpret the non-logical symbols as follows

$$0^M = \eta^M = (0, 0),$$
$$s^M((m, n)) = (m, n + 1),$$
$$f^M((m, n)) = (1, n),$$
$$P^M = \{(0, n) \mid n \in \mathbb{N}\}.$$

It is clear that M is a $\{0, s, P, f\}$ structure and moreover it is straightforward to verify that M is a model of $\mathcal{P}$. Now let us show that $M \models \mathcal{F}(\{0, s, P, f\})$-IND$_\eta$. Let $\psi(x, \vec{z})$ be a $\{0, s, P, f\}$ formula, $\vec{c}$ a vector of elements of M. Assume that $M \models \psi(0, \vec{c})$ and $M \models \psi(x, \vec{c}) \to \psi(s(x), \vec{c})$. Since $\eta^M = 0^M$, we already have $M \models \psi(\eta, \vec{c})$ and therefore $M \models I^{\eta}_{x}\psi(x, \vec{z})$. Finally, observe that $f^M(\eta^M) = (1, 0) \notin P^M$, hence $\mathcal{P} + \mathcal{F}(\{0, s, P, f\})$-IND$_\eta \nvdash P(f(\eta))$. □

The above independence result is remarkable in the sense that it imposes no restriction whatsoever on the induction formulas, only the conclusion of the induction axioms is restricted. Hence the result shows that this restriction is extremely strong. As a corollary we obtain the following unrefutability result for clause set cycles.

Corollary 49. *The $\{0, s, P, f, \eta\}$ clause set $cls(\mathcal{P} + \neg P(f(\eta)))$ is not refuted by a $\{0, s, P, f\}$ clause set cycle.*

Proof. Suppose that $cls(\mathcal{P} + \neg P(f(\eta)))$ is refuted by a $\{0, s, P, f\}$ clause set cycle. Then, by Proposition 45 we have $[\mathcal{P}, \exists_1(\{0, s, P, f\})\text{-IND}^{R-}_{\eta}] \vdash P(f(\eta))$. However, since $\mathcal{P} + \mathcal{F}(\{0, s, P, f\})$-IND$_\eta \vdash [\mathcal{P}, \exists_1(\{0, s, P, f\})\text{-IND}^{R-}_{\eta}]$, this contradicts Proposition 48. □

Lemma 50. $[\mathcal{P}, \text{Open}(\{0, s, P, f\})\text{-IND}^{R-}] \vdash P(f(\eta))$.

Proof. The formula $P(x)$ is inductive in $\mathcal{P}$. □

Proposition 48, Corollary 49, and Lemma 50 together show that the η-restriction as encountered in the n-clause calculus is drastic and can result in pathological unrefutability phenomena. On the one hand, without the η-restriction a very simple argument by induction suffices to prove $P(f(\eta))$ and on the other hand in presence of the η-restriction even induction for all $\{0, s, P, f\}$ formulas does not allow us to prove the formula $P(f(\eta))$. However, because of this the unrefutability result of Corollary 49 does not tell us anything about the other restrictions of the induction principle contained in refutations by a clause set cycle.

Hence, it would be interesting to have a similar result for linear arithmetic. In particular we conjecture the following.

Conjecture 51. $[B, \exists_1(L_{\mathrm{LA}})\text{-IND}_\eta^{R-}] \nvdash 0 + (\eta + \eta) = (\eta + \eta)$.

3.3.2. Induction rule and absence of parameters

In the following we will consider another unprovability result for clause set cycles that does not make use of the instance restriction, but instead exploits the absence of induction parameters and the induction rule. This time we work in the setting of linear arithmetic described in Section 2.2. The unprovability result developed in this section is based on the following weak cancellation property of the addition of natural numbers.

Definition 52. Let $k, n, m \in \mathbb{N}$ with $0 < n < m$, then we define

$$n \cdot x + \overline{(m-n)k} = m \cdot x \to x = \overline{k}. \tag{$\mathrm{E}_{k,n,m}$}$$

The formula $\mathrm{E}_{k,n,m}$ is a generalization of

$$x + 0 = x + x \to x = 0. \tag{$\mathrm{E}_{0,1,2}$}$$

Most of the upcoming Section 4 is devoted to proving the following independence result.

Theorem 53. *Let $n, m, k \in \mathbb{N}$ with $0 < n < m$, then*

$$(B + \mathrm{B2} + \mathrm{B3}) + \exists_1(L_{\mathrm{LA}})\text{-IND}^{R-} \nvdash \mathrm{E}_{k,n,m}.$$

By making use of the above independence result and the characterization of refutation by a clause set cycle in Proposition 45, we straightforwardly obtain an unrefutability result.

Definition 54. Let $k, n, m \in \mathbb{N}$ with $0 < n < m$, then we define the clause set $\mathcal{E}_{k,n,m}(\eta)$ by $cls(B + \mathrm{B2} + \mathrm{B3} + \neg\mathrm{E}_{k,n,m}(\eta))$.

Corollary 55. *Let $k, n, m \in \mathbb{N}$ with $0 < n < m$, then the clause set $\mathcal{E}_{k,n,m}(\eta)$ is not refuted by an L_{LA} clause set cycle.*

Proof. Assume that $cls(B + \mathrm{B2} + \mathrm{B3} + \neg\mathrm{E}_{k,n,m}(\eta))$ is refuted by a clause set cycle. By Proposition 45 we have $[B + \mathrm{B2} + \mathrm{B3}, \exists_1(L_{\mathrm{LA}})\text{-IND}_\eta^{R-}] \vdash \mathrm{E}_{k,n,m}(\eta)$. Since $(B + \mathrm{B2} + \mathrm{B3} + \exists_1(L_{\mathrm{LA}})\text{-IND}^{R-}) \vdash [B + \mathrm{B2} + \mathrm{B3}, \exists_1(L_{\mathrm{LA}})\text{-IND}_\eta^{R-}]$, this contradicts Theorem 53. □

Let us now discuss this unprovability result. The clause sets $\mathcal{E}_{k,n,m}(\eta)$ with $k, n, m \in \mathbb{N}$ and $0 < n < m$ are refuted by open induction.

Proposition 56. $\mathrm{Open}(L_{\mathrm{LA}})\text{-IND} \cup \mathcal{E}_{k,n,m}(\eta)$ *is unsatisfiable.*

Proof. Clearly, it suffices to show that $B + \mathrm{Open}(L_{\mathrm{LA}})\text{-IND} \vdash \mathrm{E}_{k,n,m}(x)$. Work in $B + \mathrm{Open}(L_{\mathrm{LA}})\text{-IND}$ and assume $n \cdot x + \overline{(m-n)k} = m \cdot x$. Then by (B2), (B3), and (B4) we obtain $\overline{(m-n)k} = (m-n) \cdot x$. Now we use (B1) to proceed by case analysis on x. If $x = \overline{k'}$ with $k' < k$, then we have $\overline{(m-n)(k-k')} = 0$. Since $m - n > 0$ and $k - k' > 0$ this contradicts Lemma 11.*(i)*. If $x = \overline{k}$, then we are done. If $x = s^{k+1}(p^{k+1}(x))$, then $0 = (m-n) + p^{k+1}(x)$, which contradicts Lemma 11.*(i)*. □

Hence, Corollary 55 together with Proposition 56 gives a positive answer to Conjecture 30. We conclude this section with some remarks on this result and possible improvements.

The formula $E_{0,1,2}(x)$ is particularly interesting, because it can be proven by a comparatively straightforward induction.

Lemma 57. $[B, \mathrm{Open}(L_{\mathrm{LA}})\text{-IND}^R] \vdash E_{0,1,2}$.

Proof. Clearly it suffices to show that the formula $\varphi(x, y) \coloneqq x + 0 = y + x \to y = 0$ is B-inductive in x. It is obvious that $B \vdash \varphi(0, y)$. Now work in B and assume $\varphi(x, y)$ and $s(x) + 0 = y + s(x)$. By (A4) and (A5) we obtain $s(x + 0) = s(x) = s(x) + 0 = y + s(x) = s(y + x)$. By Lemma 11 we obtain $x + 0 = y + x$, hence by the assumption we obtain $y = 0$. □

This demonstrates that clause set cycles are a very weak induction mechanism in the sense that they are unable to deal even with simple generalizations and therefore fail to refute relatively simple clause sets. The unprovability results in Corollaries 49 and 55 were constructed so that only one Skolem constant η appears in the language of the considered clause sets. Consider now the clause set $\mathcal{C}$ given by

$$cls(B) \cup \{\{\eta + 0 = \nu + \eta\}\} \cup \{\{\nu \neq 0\}\},$$

where ν is a Skolem constant distinct from η. It is straightforward to check that $\mathcal{C}(\eta)$ is an $L_{\mathrm{LA}} \cup \{\nu\}$ clause set cycle. Hence, if clause set cycles are detected on the languages obtained by Skolemization of the given property and its background theory, then clause set cycles allow us to prove the property $x + 0 = y + x \to y = 0$ from B but fail to prove the weaker property $x + 0 = x + x \to x = 0$ from B. Thus, clause set cycles are sensitive to the syntactic material present in a given set clauses. In particular, Skolem constants other than η may act similar to induction parameters.

The independence result of Theorem 53 also shows that the unrefutability result of Corollary 55 does neither rely on the η-restriction nor on the absence of nesting in clause set cycles. Moreover, in the light of Lemma 57 we conjecture that the unrefutability of Corollary 55 is entirely due to the absence of induction parameters from induction captured by clause set cycles.

Conjecture 58. *Let $k, n, m \in \mathbb{N}$ with $0 < n < m$, then*

$$B + \mathrm{B2} + \mathrm{B3} + \exists_1(L_{\mathrm{LA}})\text{-IND}^- \nvdash \mathrm{E}_{k,n,m}.$$

Furthermore, we believe that an independence similar to the one in Conjecture 58 also holds for the atomic formula $x + (x + x) = (x + x) + x$, which is a well-known challenging formula for inductive theorem provers [7,2,15].

Conjecture 59. $B + \exists_1(L_{\mathrm{LA}})\text{-IND}^- \nvdash x + (x + x) = (x + x) + x$.

3.3.3. Nesting of the induction rule

In this section we briefly consider the role of the depth of the nesting of applications of the induction rule. The idea underlying the results developed in this section was brought to our attention by one of the anonymous reviewers. We will show that a formalism that extends clause set cycles to achieve a fixed finite depth of the nesting of the corresponding induction rule will have an unprovable clause set, that becomes provable when the nesting depth is increased by one. Moreover, the result remains valid in extensions of clause set cycles that allow for induction parameters. However, the unprovability results in this section are more abstract than in the previous sections in the sense that we work over a much stronger background theory. We expect that providing more elementary unprovability results is not difficult but is left as future work.

In the remainder of the section we will show the following result.

Theorem 60. *Let $k \in \mathbb{N}$, then there is a language L and an $L \cup \{\eta\}$ clause set $\mathcal{C}(\eta)$ such that $\mathcal{C}$ is consistent with $[\varnothing, \exists_1(L)\text{-IND}^{R-}]_k$ but inconsistent with $[\varnothing, \exists_1(L)\text{-IND}^{R-}]_{k+1}$.*

The language of Peano arithmetic L_{PA} consists of the function symbols $0/0$, $s/1$, the infix function symbols $+/2$, $*/2$, and the infix predicate symbol $\leq /2$. Let x and y be distinct variables, then we write $(\exists x \leq y)\varphi$ as an abbreviation for $(\exists x)(x \leq y \wedge \varphi)$ and similarly we write $(\forall x \leq y)\varphi$ as an abbreviation for the formula $(\forall x)(x \leq y \to \varphi)$. A L_{PA} formula is said to be bounded if all the quantifiers occurring in it are bounded as above. The Σ_0, Π_0 and Δ_0 formulas are the bounded formulas. The Σ_{n+1} (Π_{n+1}) formulas are the formulas of the form $(\exists \vec{x})\varphi$ ($(\forall \vec{x})\varphi$) where $\vec{x}$ is a possibly empty finite sequence of variables and φ is a Π_n (Σ_n) formula.

We will prove the theorem above by providing a sequence of theories $T_0, T_1, \ldots$ with $L(T_i) \supseteq L_{\mathrm{PA}}$, $T_{i+1} = [T_i, \exists_1(L(T_0))\text{-IND}^{R-}]$ such that the provably total recursive functions of T_i are exactly those of the level $3 + i$ of the Grzegorczyk hierarchy, for $i \in \mathbb{N}$, and over T_0 the Σ_1 formulas are exactly the $\exists_1(L(T_0))$ formulas. Since, the Grzegorczyk hierarchy is a strict hierarchy (see for example [28]), we obtain for each level $i \in \mathbb{N}$ a quantifier-free $L(T_0)$ formula $\varphi(x, y)$, such that $(\exists y)\varphi(x, y)$ is provable in T_{i+1} but not in T_i.

For a definition of the Grzegorczyk hierarchy we refer the reader to [28].

Definition 61. Let $n \in \mathbb{N}$, then we denote by $\mathcal{E}_n$ the n-th level of the Grzegorczyk hierarchy.

Definition 62. The theory Q is axiomatized by the universal closure of the following axioms

$$s(x) \neq 0, \tag{Q1}$$

$$s(x) = s(y) \to x = y, \tag{Q2}$$

$$x \neq 0 \to (\exists y)(x = s(y)), \tag{Q3}$$

$$x + 0 = x, \tag{Q4}$$

$$x + s(y) = s(x + y), \tag{Q5}$$

$$x * 0 = 0, \tag{Q6}$$

$$x * s(y) = (x * y) + x, \tag{Q7}$$

$$x \leq y \leftrightarrow (\exists z)(z + x = y). \tag{Q8}$$

Definition 63. Let $n \in \mathbb{N}$, then the theory $Q + \Sigma_n$-IND is called $I\Sigma_n$. The theory $I\Sigma_0$ is also called $I\Delta_0$.

There is a Δ_0 definition of the exponential function such that the theory $I\Delta_0$ proves the inductive properties of the definition of the exponential function, but $I\Delta_0$ does not prove the totality of such a definition.

Lemma 64. *There is a Δ_0 formula $\mathrm{Exp}(x, y, z)$ such that $I\Delta_0$ proves*

$$\mathrm{Exp}(x, 0, z) \leftrightarrow z = \overline{1}, \tag{E1}$$

$$\mathrm{Exp}(x, s(y), z) \leftrightarrow (\exists v)(\mathrm{Exp}(x, y, v) \wedge z = v * x). \tag{E2}$$

In particular $I\Delta_0$ proves $\mathrm{Exp}(x, y, z_1) \wedge \mathrm{Exp}(x, y, z_2) \to z_1 = z_2$.

Proof. See [17, Section V.3] □

In the following we will mainly work a theory that extends $I\Delta_0$ by a statement asserting the totality of the exponential function.

Definition 65. By $I\Delta_0 + \mathrm{EXP}$ we denote the theory that extends $I\Delta_0$ by the axiom $(\forall x)(\forall y)(\exists z)\mathrm{Exp}(x, y, z)$.

The theory $I\Delta_0 + \mathrm{EXP}$ is also called elementary arithmetic and has various equivalent formulations, see [6, Section 1.1]. In the following we will develop a particular formulation with a $\forall_1$ axiomatization and in which the $\exists_1$ formulas of the extended language are exactly the Σ_1 formulas.

Lemma 66. *$I\Delta_0$ has a Π_1 axiomatization.*

Proof. Drop axiom (Q3), replace axiom (Q8) by the universal closure of the formulas $x \leq y \to (\exists z \leq y)z + x = y$ and $z + x = y \to x \leq y$, and replace the induction axioms $I_x\varphi$, where $\varphi(x, \vec{z})$ is Δ_0 by

$$\big(\varphi(0, \vec{z}) \wedge (\forall y < x)\big(\varphi(y, \vec{z}) \to \varphi(s(y), \vec{z})\big)\big) \to \varphi(x, \vec{z}).$$

It is routine to check that the resulting theory is equivalent to $I\Delta_0$. □

Now we will show that $I\Delta_0$ has Δ_0 definitions of Skolem functions of all Δ_0 formulas. Later on we will introduce the corresponding Skolem functions in order to get rid of bounded quantifiers.

Definition 67 *(Least number principle).* Let $\varphi(x, \vec{z})$ be a formula, then the least number principle for φ is given by

$$(\exists x)\varphi(x, \vec{z}) \to (\exists x)\big(\varphi(x, \vec{z}) \wedge (\forall y < x)\neg\varphi(y, \vec{z})\big).$$

Lemma 68. *$I\Delta_0$ proves the least number principle for Δ_0 formulas.*

Proof. See [17, Theorem 1.22]. □

Definition 69. Let $\varphi(\vec{x}, y, z)$ be a Δ_0 formula, then the formula $D_{(\exists z \leq y)\varphi}(\vec{x}, y, z)$ is given by

$$\big(z \leq y \wedge \varphi(x, y, z) \wedge (\forall z' < z)\neg\varphi(\vec{x}, y, z')\big) \vee \big((\forall z \leq y)\neg\varphi(\vec{x}, y, z) \wedge z = 0\big).$$

Lemma 70. *Let $\varphi(\vec{x}, y, z)$ be a Δ_0 formula, then $I\Delta_0$ proves*

(i) $(\exists z \leq y)\varphi(\vec{x}, y, z) \to (\exists z \leq y)(D_{(\exists z \leq y)\varphi}(\vec{x}, y, z) \wedge \varphi(\vec{x}, y, z))$
(ii) $(\forall z \leq y)\neg\varphi(\vec{x}, y, z) \to D_{(\exists z \leq y)\varphi}(\vec{x}, y, 0)$
(iii) $(\exists! z)D_{(\exists z \leq y)\varphi}(\vec{x}, y, z)$,
(iv) $D_{(\exists z \leq y)\varphi}(\vec{x}, y, z) \to z \leq y$.

Proof. The formula 70.*(i)* follows easily from Lemma 68. The formulas 70.*(ii)*–70.*(iv)* are straightforward. □

We will now define the way in which we Skolemize Δ_0 formulas.

Definition 71. Let $\varphi(\vec{x}, y, z)$ be a Δ_0 formula, then $F_{(\exists z \leq y)\varphi}$ is a function symbol of arity $|\vec{x}| + 1$.

Definition 72. The formula translations $(\cdot)^{\exists}$ and $(\cdot)^{\forall}$ are defined mutually recursively by

$$
\begin{aligned}
&(\theta)^{Q} = \theta, \text{ if } \theta \text{ is quantifier-free},\\
&(\varphi_1 \wedge \varphi_2)^{Q} = \varphi_1^{Q} \wedge \varphi_2^{Q},\\
&(\varphi_1 \vee \varphi_2)^{Q} = \varphi_1^{Q} \vee \varphi_2^{Q},\\
&(\neg\varphi)^{Q} = \neg\varphi^{\overline{Q}},\\
&((\overline{Q}\, y \leq x)\varphi)^{Q} = (\overline{Q}\, y \leq x)\varphi^{Q}\\
&((\exists y \leq x)\varphi(x, y, \vec{z}))^{\exists} = \left(y \leq x \wedge \varphi^{\exists}\right)[y/F_{(\exists y \leq x)\varphi}(x, \vec{z})], \qquad (*_1)\\
&((\forall y \leq x)\varphi(x, y, \vec{z}))^{\forall} = \left(y \leq x \to \varphi^{\forall}\right)[y/F_{(\exists y \leq x)\neg\varphi}(x, \vec{z})], \qquad (*_2)
\end{aligned}
$$

where $Q \in \{\forall, \exists\}$, $\overline{\forall} = \exists$, $\overline{\exists} = \forall$, and in Equations $(*_1)$ and $(*_2)$ the variables $\vec{z}$ all appear freely in the formula φ.

We can now obtain a suitable formulation of $I\Delta_0 + \mathrm{EXP}$.

Lemma 73. *There exists a $\forall_1$ axiomatized conservative extension T of $I\Delta_0 + \mathrm{EXP}$ such that every $\exists_1(L(T))$ formula is equivalent over T to a Σ_1 formula and every Σ_1 formula is equivalent over T to an $\exists_1(L(T))$ formula.*

Proof. We consider a Π_1 formulation U of $I\Delta_0$. For each axiom $(\forall\vec{x})\varphi$ of U where φ is Δ_0, T contains the axiom $(\forall\vec{x})\varphi^{\exists}$. Furthermore, T contains the axiom $\mathrm{Exp}^{\exists}[z/e(x, y)]$. Finally, for each Δ_0 formula $\varphi(\vec{x}, y, z)$, T contains the axiom $(D_{(\exists z \leq y)\varphi})^{\exists}[z/F_{(\exists z \leq y)\varphi}(\vec{x}, y)]$. Now obtain a $\forall_1$ axiomatization by moving the remaining quantifiers outwards. By a model-theoretic argument it is straightforward to see that the resulting theory is conservative over $I\Delta_0 + \mathrm{EXP}$.

It is straightforward to check that every Δ_0 formula φ is equivalent in T to a quantifier-free $L(T)$ formula. Let ψ be a Σ_1 formula, then $\psi = (\exists\vec{x})\varphi$ where φ is Δ_0. Hence, ψ is equivalent over T to the formula $(\exists\vec{x})\varphi'$ where φ' is a quantifier-free formula that is equivalent over T to φ. Now let ψ be an $\exists_1(L(T))$ formula, then by [16, pp. 51–52] there exists an equivalent unnested $\exists_1(L(T))$ formula of the form $(\exists\vec{x})\varphi$ where φ is quantifier-free. Now we simply replace atoms of the form $f(\vec{u}) = y$ where f is either a Skolem symbol of a Δ_0 formula or e by the corresponding defining Δ_0 formula. Hence, the resulting formula is a Σ_1 formula. □

In the following we fix one such extension of $I\Delta_0 + \mathrm{EXP}$ and call it EA.

Definition 74. Let $k \in \mathbb{N}$, then EA_k denotes the theory $[\mathrm{EA}, \Pi_2\text{-}\mathrm{IND}^R]_k$.

Theorem 75 *([32]). The provably total recursive functions of the theory EA_k are precisely those of the class $\mathcal{E}_{3+k}$ of the Grzegorczyk hierarchy.*

Proof. See also the proof Corollary 7.5 of [3]. □

We can reformulate the theories EA_k as follows.

Lemma 76. *Let $k \in \mathbb{N}$, then $\mathrm{EA}_k \equiv [\mathrm{EA}, \exists_1(L(\mathrm{EA}))\text{-}\mathrm{IND}^{R-}]_k$.*

Proof. We proceed by induction on k and show

$$\mathrm{EA}_k \equiv [\mathrm{EA}, \exists_1(L(\mathrm{EA}))\text{-}\mathrm{IND}^{R-}]_k.$$

If $k = 0$, then the claim follows trivially. Now assume the claim for k, then EA_k is Π_2 axiomatized, hence by [3, Corollary 7.4]

$$\mathrm{EA}_{k+1} \equiv [\mathrm{EA}_k, \Pi_2\text{-}\mathrm{IND}^R] \equiv [\mathrm{EA}_k, \Sigma_1\text{-}\mathrm{IND}^R].$$

Furthermore, by [6, Lemma 4.6] we have

$$[\mathrm{EA}_k, \Sigma_1\text{-}\mathrm{IND}^R] \equiv [\mathrm{EA}_k, \Sigma_1\text{-}\mathrm{IND}^{R-}].$$

Since over EA the Σ_1 formulas are exactly the $\exists_1(L(\mathrm{EA}))$ formulas, we obtain

$$[\mathrm{EA}_k, \Sigma_1\text{-IND}^{R-}] \equiv [\mathrm{EA}_k, \exists_1(L(\mathrm{EA}))\text{-IND}^{R-}].$$

By the induction hypothesis we readily obtain

$$[\mathrm{EA}, \Pi_2\text{-IND}^{R}]_{k+1} \equiv [\mathrm{EA}, \exists_1(L(\mathrm{EA}))\text{-IND}^{R-}]_{k+1}. \quad \square$$

Since $\mathcal{E}_k \subsetneq \mathcal{E}_{k+1}$ for all $k \in \mathbb{N}$, we can now provide a proof of Theorem 60.

Proof of Theorem 60. Let $k \in \mathbb{N}$, then there exists a function $f : \mathbb{N} \to \mathbb{N}$ such that $f \in \mathcal{E}_{k+4} \setminus \mathcal{E}_{k+3}$. Hence, there exists a Σ_1 formula $\varphi(x, y)$ such that $f(n) = m$ if and only if $\mathbb{N} \models \varphi(\overline{n}, \overline{m})$ and

$$[\mathrm{EA}, \exists_1(L(\mathrm{EA}))\text{-IND}^{R-}]_{k+1} \vdash (\exists y)\varphi(x, y),$$

$$[\mathrm{EA}, \exists_1(L(\mathrm{EA}))\text{-IND}^{R-}]_{k} \nvdash (\exists y)\varphi(x, y).$$

Thus, by the construction of EA, there exists a quantifier-free $L(\mathrm{EA})$ formula $\varphi'(x, y, \vec{z})$ such that $\mathrm{EA} \vdash \varphi \leftrightarrow (\exists \vec{z})\varphi'$. Since EA is $\forall_1$ axiomatized we furthermore have $\mathrm{EA} + [\varnothing, \exists_1(L(\mathrm{EA}))\text{-IND}^{R-}] \equiv [\mathrm{EA}, \exists_1(L(\mathrm{EA}))\text{-IND}^{R-}]$. Hence, $\mathcal{C} \coloneqq cls(\mathrm{EA} + (\forall y)(\forall \vec{z})\neg\varphi'(\eta, y, \vec{z}))$ is a suitable clause set. $\square$

This result tells us that a mechanism that extends refutation by a clause set cycle so as to allow at most k-fold nested $\exists_1$ parameter-free induction rule is strictly weaker than a mechanism that allows $(k+1)$-fold nested applications of the $\exists_1$ parameter-free induction rule. This naturally gives rise to the question whether we can separate a system that provides arbitrary nestings of the parameter-free $\exists_1$ induction rule from a system that provides the parameter-free $\exists_1$ induction schema. The following lemma shows that we need a different approach to resolve this question.

Lemma 77 *([26]). $I\Sigma_1$ is Π_2 conservative over $\mathrm{EA} + \Sigma_1\text{-IND}^R$.*

Hence the theory $\mathrm{EA} + \exists_1(L(\mathrm{EA}))\text{-IND}$ is also Π_2 conservative over $\mathrm{EA} + L(\mathrm{EA})\text{-IND}^{R-}$. Thus the technique used above does not provide us with a clause set that separates both systems.

Nevertheless, we conjecture that the parameter-free $\exists_1$ induction schema is in general stronger than the parameter-free $\exists_1$ induction rule.

Conjecture 78. *There exists a language L and an $L \cup \{\eta\}$ clause set $\mathcal{D}(\eta)$ such that $\exists_1(L)\text{-IND}^- + \mathcal{D}(\eta)$ is inconsistent, but $(\varnothing + \exists_1(L)\text{-IND}^{R-}) + \mathcal{D}(\eta)$ is consistent.*

The results in this section are less elementary than the results of Sections 3.3.1 and 3.3.2 in the sense that we work over the comparatively strong EA and the separation involves clause sets that express totality assertions. However, totality assertions are an important class of problems for AITP systems. In this sense the connection with the Grzegorczyk hierarchy is remarkable.

4. Idempotents in linear arithmetic

In the previous section we have introduced clause set cycles and we have given a characterization of refutation by a clause set cycle in terms of a logical theory. Moreover, we have shown two unrefutability results for clause set cycles. We have shown the second unrefutability result by anticipating the independence result of Theorem 53 for which a proof will be provided in this section. In Section 4.1 we introduce some preliminary notions and we carry out some syntactic simplifications on $\exists_1$ formulas. In Section 4.2 we consider some properties of $\exists_1$ formulas in the structures $\mathbb{N}$ and $\mathbb{Z}$. Finally, in Section 4.3 we carry out the model theoretic construction.

We work in the setting of linear arithmetic, hence, unless stated otherwise, whenever we speak of a formula (sentence) we mean an L_{LA} formula (sentence).

4.1. Preliminaries

In this section we mainly carry out some syntactic transformations that allow us to eliminate the function symbols p and 0 from $\exists_1$ formulas. The absence of these symbols allows us to carry out certain embeddings of structures in Sections 4.2 and 4.3.

Definition 79. The theory $\mathcal{V}$ is axiomatized by the universal closure of the formulas

$$\overline{k} + x = x + \overline{k}, \tag{$\mathrm{V_k}$}$$

where $k \in \mathbb{N}$.

Lemma 80. $[B, \mathrm{Open}(\{0, s, +\})\text{-IND}^{R-}] \vdash \mathcal{V}$.

Proof. The formula $\overline{k} + x = s^k(x)$ is B-inductive and furthermore $B \vdash s^k(x) = x + \overline{k}$. □

We will carry out these transformations in the very weak theory $B + \mathrm{B1} + \mathcal{V}$. In a first step we will show that we can eliminate the symbol p from $\exists_1$ formulas without increasing the quantifier complexity of $\exists_1$ formulas. After that, we show that we can moreover eliminate to a certain extent the symbol 0 from $\exists_1$ formulas, again without increasing the quantifier complexity.

In order to eliminate the symbol p from $\exists_1$ formulas we proceed by replacing all the occurrences of the symbol p by the following definition of the predecessor function.

Definition 81. We define the formula $\mathrm{D}(x, y)$ by

$$(x = 0 \wedge y = 0) \vee s(y) = x.$$

Lemma 82. $B + \mathrm{B1} \vdash p(x) = y \leftrightarrow \mathrm{D}(x, y)$.

Proof. We work in $B + \mathrm{B1}$. Assume $p(x) = y$. If $x = 0$, then we have $y = p(x) = p(0) = 0$, hence $\mathrm{D}(x, y)$. Otherwise, $x = s(p(x))$ and therefore $x = s(p(x)) = s(y)$. Now assume $\mathrm{D}(x, y)$. If $x = 0 \wedge y = 0$, then we have $p(x) = p(0) = 0 = y$. If $s(y) = x$, then $y = p(s(y)) = p(x)$. □

We can now factor the symbol p into the axiom B1 by replacing all the occurrences of p by the definition of the predecessor function.

Lemma 83. *Let $\varphi(\vec{x})$ be an $\exists_1$ L_{LA} formula, then there exists a p-free $\exists_1$ formula $\varphi'(\vec{x})$ such that*

$$B + \mathrm{B1} \vdash \varphi \leftrightarrow \varphi'.$$

Proof. Let φ be an $\exists_1(L_{\mathrm{LA}})$ formula, then there exists an unnested $\exists_1(L_{\mathrm{LA}})$ formula ψ such that $\vdash \varphi \leftrightarrow \psi$, see for example [16, pp. 51–52]. In particular, the symbol p occurs in ψ only in atoms of the form $p(x) = y$. Hence, we obtain the desired formula by replacing in ψ the atomic formulas of the form $p(x) = y$ by $\mathrm{D}(x, y)$. □

In the following we will eliminate the symbol 0 to a certain extent from $\exists_1$ formulas in one variable. In order to simplify the arguments we will introduce some additional assumptions. Since we work in the context of the theory B we can by Lemma 13 assume without loss of generality that ground terms are numerals. Moreover, since equality is symmetric we will assume without loss of generality that atoms are oriented in such a way that whenever the atom contains a variable, then the left hand side of the atom contains a variable.

Let us start by introducing the notion of components, a class of $\exists_1$ formulas that is particularly suitable to carry out the elimination of the symbol 0. Moreover, components will also be of use for the arguments in Section 4.2.

Definition 84 *(Components).* A component $\chi(\vec{x})$ is a formula of the form $\exists \vec{y} C_\chi(\vec{x}, \vec{y})$, where C_χ is a conjunction of literals.

Lemma 85. *Let $\varphi(x)$ be an $\exists_1$ formula, then there exist p-free components $\chi_1, \ldots, \chi_n$ such that $B + \mathrm{B1} \vdash \varphi \leftrightarrow \bigvee_{i=1}^n \chi_i$.*

Proof. Apply Lemma 83 to obtain a p-free $\exists_1$ formula φ' such that $B + \mathrm{B1} \vdash \varphi \leftrightarrow \varphi'$. Now obtain the desired components by replacing formulas of the form $\varphi \to \psi$ and $\varphi \leftrightarrow \psi$ respectively by $\neg\varphi \vee \psi$ and $(\neg\varphi \vee \psi) \wedge (\neg\psi \vee \varphi)$, moving negations inward, eliminating double negations, distributing conjunctions over disjunctions, and moving existential quantifiers inwards over disjunctions. □

We will distinguish between three types of literals: Those where both sides contain variables, those where only one side of the equation contains a variable and those where none of the sides contain a variable.

Definition 86. Let l be a literal of the form $u \bowtie v$ with $\bowtie \in \{=, \neq\}$, then l is: $\uparrow\uparrow$ if both u and v contain a variable, $\uparrow\downarrow$ if u contains a variable and v is ground, and $\downarrow\downarrow$ if both u and v are ground. We will combine this notation with superscript $+$ to indicate that the literal is positive and a superscript $-$ to indicate that the literal is negative. We say that a $\uparrow\downarrow$ literal is simple if it is of the form $z = \overline{k}$ where z is a variable and $k \in \mathbb{N}$ and complex otherwise.

Lemma 87. *Let t be a term with $\mathrm{Var}(t) \neq \varnothing$, then there exists a 0-free term t' such that $B + \mathcal{V} \vdash t = t'$.*

Proof. We proceed by induction on the structure of the term t. If t is a variable, then we are done by letting $t' = t$. If t is of the form $s(u)$, then $Var(u) \neq \varnothing$. Hence, we can apply the induction hypothesis to u in order to obtain a 0-free term u' such that $B + \mathcal{V} \vdash u = u'$. Thus, $B + \mathcal{V} \vdash t = s(u')$ and we let $t' = s(u')$. If t is of the form $p(u)$, then we proceed analogously. If t is of the form $u + v$, then we need to consider several cases. If $Var(u) = \varnothing$, then $Var(v) \neq \varnothing$ and we have $B \vdash u = \overline{k}$ for some $k \in \mathbb{N}$ and therefore $B + \mathcal{V} \vdash t = \overline{k} + v = v + \overline{k} = v' + \overline{k} = s^k(v')$. If $Var(v) = \varnothing$, then $Var(u) \neq \varnothing$. Hence, we can apply the induction hypothesis to u in order to obtain a 0-free term u' such that $B + \mathcal{V} \vdash u = u'$. Since $Var(v) = \varnothing$, there exists $k \in \mathbb{N}$ such that $B \vdash v = \overline{k}$. By multiple applications of (A5) followed by an application of (A5) we obtain $B + \mathcal{V} \vdash t = u + \overline{k} = s^k(u) + 0 = s^k(u)$. Hence, $t' \coloneqq s^k(u)$ is the desired 0-free term. If u and v contain variables, then by the induction hypothesis we obtain 0-free terms u' and v' such that $B + \mathcal{V} \vdash u = u'$ and $B + \mathcal{V} \vdash v = v'$. Hence, $t = u' + v'$ is the desired 0-free term. □

By Lemma 14 and Lemma 87, it is straightforward to eliminate the symbol 0 from $\uparrow\uparrow$ and $\downarrow\downarrow$ literals. However, eliminating the symbol 0 from $\uparrow\downarrow$ literals needs some more work. Let us start by observing that complex $\uparrow\downarrow$ atoms can be split into several simple ones.

Lemma 88. *Let $u(\vec{z})$ be a p-free term with $\vec{z} = (z_1, \dots, z_l)$ and $k \in \mathbb{N}$, then*

$$B + \mathrm{B1} \vdash u(\vec{z}) = \overline{k} \leftrightarrow \bigvee_{\substack{0 \leq m_1, \dots, m_l \leq k \\ \mathbb{N} \models u(m_1, \dots, m_l) = k}} \bigwedge_{j=1}^{l} z_j = \overline{m_j}.$$

Proof. Work in $B + \mathrm{B1}$. The "if" direction is obvious. For the "only if" direction assume $u(\vec{z}) = \overline{k}$ and proceed by k-fold case analysis on the variables $\vec{z}$. If $z_i = \overline{m_i}$ with $0 \leq m_i \leq k$ for $i = 1, \dots, l$, then we have two cases. If $u(\overline{m_1}, \dots, \overline{m_l}) \neq \overline{k}$, then the claim follows trivially. Otherwise if $u(\overline{m_1}, \dots, \overline{m_l}) = \overline{k}$, then we are done as well since $z_1 = \overline{m_1} \wedge \dots \wedge z_l = \overline{m_l}$ is a conjunct of the right side. Otherwise, there exists an $i \in \{1, \dots, l\}$ and z_i' such that $z_i = s^{k+1} z_i'$. Then let j be the index of the variable z_j with the rightmost occurrence such that $z_j = s^{k+1} z_j'$ for some z_j'. Then we have $u(\vec{z}) = s^{k+1}(u'(z_1, \dots, z_{j-1}, z_j', z_{j+1}, \dots, z_l))$ and a term u'. Hence, by Lemma 11.*(i)* we have $u(\vec{z}) \neq \overline{k}$. □

Furthermore, we can eliminate simple $\uparrow\downarrow^-$ literals at the expense of introducing several positive literals and an existential quantifier.

Lemma 89. *Let $k \in \mathbb{N}$, then*

$$B + \mathrm{B1} \vdash z \neq \overline{k} \leftrightarrow \left((\exists z') z = s^{k+1} z' \vee \bigvee_{i=0}^{k-1} z = \overline{i} \right).$$

Proof. The "if" direction is obvious. For the "only if" direction assume $z \neq \overline{k}$ and proceed by k-fold case analysis on z. If $z = \overline{i}$ with $i < k$, then we are done. The case where $z = \overline{k}$ contradicts the assumption and therefore we are done. If $z = s^{k+1} z'$ for some z', then we are done as well. □

The elimination of the $\uparrow\downarrow$ literals from a component $\chi(x_1, \dots, x_m)$ consists of two majors steps. In a first step we deal with all the $\uparrow\downarrow$ literals except the simple $\uparrow\downarrow$ literals of the form $x_i = \overline{k}$ with $k \in \mathbb{N}$ and $i \in \{1, \dots, m\}$. In the second step we will deal with the remaining $\uparrow\downarrow$ literals by making use of the observation that the truth value of a literal of the form $x = \overline{k}$ with $k \in \mathbb{N}$ becomes fixed when x is large enough.

Let us start by defining some measures that will be used to control the first step of the elimination procedure.

Definition 90. Let $\chi(\vec{x}) = (\exists y_1) \dots (\exists y_l) C_\chi$ be a component, then $\#^-(\chi)$ is the number of occurrences of negative literals in χ, $\#_\exists(\chi) = l$, $\#^+_{\text{complex}}(\chi)$ is the number of occurrences of complex $\uparrow\downarrow^+$ literals in χ, and $\#_{\mathrm{FV}}(\chi)$ is the number of free variables of χ.

We will now provide some intermediate lemmas that allow us to eliminate a single literal.

Lemma 91 *(Elimination of $\uparrow\downarrow^-$ literals). Let $\chi(\vec{x})$ be a p-free component containing a $\uparrow\downarrow^-$ literal, then there exist p-free components $\chi_1', \dots, \chi_n'$ such that $B + \mathrm{B1} \vdash \chi \leftrightarrow \bigvee_{i=1}^n \chi_i'$ and $\#^-(\chi_i') < \#^-(\chi)$ for $i = 1, \dots, n$.*

Proof. We first apply Lemma 88 in order to split the atom of the $\uparrow\downarrow^-$ literal. After that, we move the negations inwards and apply Lemma 89 to all the newly introduced literals of the form $z \neq \overline{k}$ with $k \in \mathbb{N}$. Now we move the newly introduced existential quantifiers outwards and possibly rename some bound variables. Finally, we distribute conjunctions over

disjunctions exhaustively. Let $\chi_1, \ldots, \chi_k$ be the resulting components. Since we have introduced only existential quantifiers and positive literals, we have $\#^-(\chi_i) < \#^-(\chi)$. □

Lemma 92 *(Elimination of complex $\uparrow\downarrow^+$ literals). Let $\chi(\vec{x})$ be a p-free component containing a complex $\uparrow\downarrow^+$ literal, then there exist p-free components $\chi'_1, \ldots, \chi'_n$ with $B + \mathrm{B1} \vdash \chi \leftrightarrow \bigvee_{i=1}^n \chi'_i$ such that $\#^-(\chi'_i) = \#^-(\chi)$, $\#_\exists(\chi'_i) = \#_\exists(\chi)$, and $\#^+_{\mathrm{complex}}(\chi'_i) < \#^+_{\mathrm{complex}}(\chi)$ for $i = 1, \ldots, n$.*

Proof. We apply Lemma 88 to split a complex $\uparrow\downarrow^+$ literal. Now obtain components $\chi'_1, \ldots, \chi'_n$ by distributing conjunctions over disjunctions exhaustively and moving the existential quantifiers inwards over the disjunctions. Clearly we have $B + \mathrm{B1} \vdash \chi \leftrightarrow \bigvee_{i=1}^n \chi'_i$. Observe that this operation does not introduce any negative literals or quantifiers. Hence we have $\#^-(\chi'_i) = \#(\chi)$ and $\#_\exists(\chi'_i) = \#_\exists(\chi)$ for $i = 1, \ldots, n$. Moreover, only simple $\uparrow\downarrow^+$ literals are introduced, hence we have $\#^+_{\mathrm{complex}}(\chi'_i) < \#^+_{\mathrm{complex}}(\chi)$ for $i = 1, \ldots, n$. □

Lemma 93 *(Elimination of simple $\uparrow\downarrow^+$ literals). Let $\chi(\vec{x}) = (\exists y_1)\ldots(\exists y_l)C_\chi$ be a p-free component containing a literal of the form $y_i = \overline{k}$, then there exists a p-free component $\chi'(x)$ such that $\vdash \chi \leftrightarrow \chi'$, $\#^-(\chi') = \#^-(\chi)$, and $\#_\exists(\chi') < \#_\exists(\chi)$.*

Proof. Let us assume without loss of generality that $C = y_i = \overline{k} \wedge C'(\vec{x}, y_1, \ldots, y_l)$, where C' is a conjunction of literals. Then, it suffices to apply the first-order equivalence

$$\vdash (\exists y_i)\Big(y_i = \overline{k} \wedge C'(\vec{x}, y_1, \ldots, y_{i-1}, y_i, y_{i+1}, \ldots, y_l)\Big) \leftrightarrow C'(\vec{x}, y_1, \ldots, y_{i-1}, \overline{k}, y_{i+1}, \ldots, y_l).$$

Clearly we have $\#^-(\chi') = \#^-(\chi)$ and $\#_\exists(\chi') < \#_\exists(\chi)$. □

The following lemma combines the previous lemmas in order to accomplish the first step of the elimination of the $\uparrow\downarrow$ literals.

Lemma 94. *Over $B + \mathrm{B1} + \mathcal{V}$ every $\exists_1(L_{\mathrm{LA}})$ formula $\varphi(x_1, \ldots, x_n)$ is equivalent to a disjunction of formulas of the form $\bigwedge_{i\in I} x_i = \overline{k_i} \wedge (\exists\vec{y})C(\vec{x}, \vec{y})$, where $I \subseteq [n] = \{1, \ldots, n\}$ and C is a p-free 0-free conjunction of literals that contains only those variables x_i such that $i \notin I$.*

Proof. Let $\chi(\vec{x})$ be a p-free component, then we proceed by induction on the lexicographic order $\prec$ on $\mathbb{N}^4$ induced by $\leq$ and show that over $B + \mathrm{B1}$ the component χ is equivalent to disjunction of formulas of the form $\bigwedge_{i\in I} x_i = \overline{k_i} \wedge (\exists\vec{y})C(\vec{x}, \vec{y})$, where $I \subseteq [n]$ and C is a p-free disjunction of $\uparrow\uparrow$ and $\downarrow\downarrow$ literals that contains only those variables x_i such that $i \notin I$. Let $\#(\chi) = (\#^-(\chi), \#_\exists(\chi), \#^+_{\mathrm{complex}}(\chi), \#_{\mathrm{FV}}(\chi))$. If χ contains a $\uparrow\downarrow^-$ literal, then we apply Lemma 91 in order to obtain p-free components $\chi'_1, \ldots, \chi'_n$ such that $B + \mathrm{B1} \vdash \chi \leftrightarrow \bigvee_{i=1}^n \chi'_i$ and $\#^-(\chi'_i) < \#^-(\chi)$. Hence $\#(\chi'_i) \prec \#(\chi)$ and therefore we can apply the induction hypothesis to each of $\chi'_1, \ldots, \chi'_n$ in order to obtain the desired components. If χ contains a complex $\uparrow\downarrow^+$ literal, then we apply Lemma 92 in order to obtain p-free components $\chi'_1, \ldots, \chi'_n$ with $B + \mathrm{B1} \vdash \chi \leftrightarrow \bigvee_{i=1}^n \chi'_i$ such that $\#^-(\chi'_i) = \#^-(\chi)$, $\#_\exists(\chi'_i) = \#_\exists(\chi)$ and $\#^+_{\mathrm{complex}}(\chi'_i) < \#^+_{\mathrm{complex}}(\chi)$, for $i = 1, \ldots, n$. Hence $\#(\chi'_i) \prec \#(\chi)$ for $i = 1, \ldots, n$ and therefore we can apply the induction hypothesis to $\chi'_1, \ldots, \chi'_n$ in order to obtain the desired components. Let $\chi(x) = (\exists y_1)\ldots(\exists y_l)C_\chi$. If χ contains a $\uparrow\downarrow$ literal of the form $x_i = \overline{k_i}$ with $i \in \{1, \ldots, n\}$, then let $\chi = (\exists\vec{y})C_\chi(\vec{x}, \vec{y})$ and $\chi' = (\exists\vec{y})C_\chi[x_i/\overline{k_i}]$. We have $\vdash \chi \leftrightarrow x_i = \overline{k_i} \wedge \chi'$. Clearly, $\#^-(\chi') = \#^-(\chi)$, $\#_\exists(\chi') = \#_\exists(\chi)$, and $\#^+_{\mathrm{complex}}(\chi') \leq \#^+_{\mathrm{complex}}(\chi)$ but $\#_{\mathrm{FV}}(\chi') = \#_{\mathrm{FV}}(\chi) - 1$. Hence, we may apply the induction hypothesis to the component χ'. If χ contains a simple $\uparrow\downarrow^+$ literal $y_i = \overline{k}$, then we apply Lemma 93 in order to obtain a p-free component $\chi'(x)$ such that $B + \mathrm{B1} \vdash \chi \leftrightarrow \chi'$, $\#^-(\chi') = \#^-(\chi)$, and $\#_\exists(\chi') < \#_\exists(\chi)$. Hence we have $\#(\chi') \prec \#(\chi)$ and therefore we can apply the induction hypothesis in order to obtain the desired components.

Now let $\varphi(x_1, \ldots, x_n)$ be an $\exists_1(L_{\mathrm{LA}})$ formula. By Lemma 85 the formula φ is equivalent over $B + \mathrm{B1}$ to a disjunction of p-free components. Therefore, by the procedure above the formula φ is equivalent over $B + \mathrm{B1}$ to a disjunction of formulas of the form $\bigwedge_{i\in I} x_i = \overline{k_i} \wedge (\exists\vec{y})C(\vec{x}, \vec{y})$, where $I \subseteq [n]$ and C is a p-free disjunction of $\uparrow\uparrow$ and $\downarrow\downarrow$ literals containing only those variables x_i such that $i \notin I$. Now we apply Lemma 14 to eliminate the $\downarrow\downarrow$ literals from C and Lemma 87 to eliminate 0 from the $\uparrow\uparrow$ literals of C. □

In the next step we eliminate the remaining literals of the form $x = \overline{k}$. This step relies on the observation that the truth value of these literals is fixed when x is large enough.

Proposition 95. *Let $\varphi(x_1, \ldots, x_n)$ be an $\exists_1$ formula, then there exists $N \in \mathbb{N}$ such that $\varphi(s^N(x_1), \ldots, s^N(x_n))$ is equivalent over $B + \mathrm{B1} + \mathcal{V}$ to a 0-free, p-free, $\exists_1$ formula.*

Proof. By Lemma 94 the formula φ is equivalent over $B + \mathrm{B1} + \mathcal{V}$ to a disjunction of the form

$$\bigvee_{j=1}^{m} \left(\bigwedge_{i \in I_j} x_i = \overline{k}_{j,i} \wedge (\exists \vec{y}) C_j(\vec{x}, \vec{y}) \right),$$

where for $j = 1, \ldots, m$, $I_j \subseteq [n]$ and C_j is a p-free 0-free disjunction of literals containing only those variables x_i such that $i \notin I_j$. Let $N = 1 + \max\{k_{i,j} \mid j = 1, \ldots, m, i \in I_j\}$, then $\varphi(s^N(x_1), \ldots, s^N(x_n))$ is equivalent over $B + \mathrm{B1} + \mathcal{V}$ to the formula

$$\bigvee_{\substack{j=1,\ldots,m \\ I_j = \varnothing}} (\exists \vec{y}) C_j(\vec{x}, \vec{y}).$$

Finally, we obtain the desired formula by moving the $\exists$ quantifiers outwards over the disjunction. □

4.2. Components in $\mathbb{N}$ and $\mathbb{Z}$

In this section we will investigate some basic model-theoretic properties of $\exists_1$ formulas in the structures $\mathbb{N}$ and $\mathbb{Z}$.

Definition 96. Let M be an L_{LA} structure and $\varphi(x)$ a formula. We say that $d \in M$ is a solution of φ in M if $M \models \varphi(d)$.

We will show that every p-free $\exists_1$ formula with enough solutions in $\mathbb{N}|_{\{0,s,+\}}$ has an infinite strictly descending sequence of solutions in $\mathbb{Z}|_{\{0,s,+\}}$. Since the structure $\mathbb{N}|_{\{0,s,+\}}$ can be embedded into $\mathbb{Z}|_{\{0,s,+\}}$, it is clear that if $\mathbb{N}|_{\{0,s,+\}} \models \varphi(n)$, then $\mathbb{Z}|_{\{0,s,+\}} \models \varphi(n)$, for all $n \in \mathbb{N}$.

Let $\theta(x_1, \ldots, x_k)$ be an atom, then it is obvious that θ is equivalent in $\mathbb{Z}$ to a linear equation of the form $\sum_{i=1}^{k} a_i x_i = b$ with integers $a_1, \ldots, a_k, b$. Hence a conjunction of atoms $\theta_1(x_1, \ldots, x_k), \ldots, \theta_n(x_1, \ldots, x_k)$ is equivalent over $\mathbb{Z}$ to an inhomogeneous system of linear equations of the form

$$A\vec{x} = b, \tag{I}$$

where $A \in \mathbb{Z}^{m \times k}$ and $b \in \mathbb{Z}^{m \times 1}$. Now consider the corresponding homogeneous system

$$A\vec{x} = \mathbf{0}. \tag{H}$$

The solutions of the system (H) form a submonoid $\mathcal{H}$ of $\mathbb{Z}^k$ with pointwise addition. Furthermore, assume that (I) has a particular solution $i_{(p)}$, then the set of solutions of (I) is given by

$$\mathcal{I} = \{h + i_{(p)} \mid h \in \mathcal{H}\}.$$

Lemma 97. *Let $\chi(x)$ be a component with two solutions in $\mathbb{Z}$, then for all $n \in \mathbb{N}$ there exists $n' \in \mathbb{N}$ with $n' \geq n$ such that $\mathbb{Z} \models \chi(-n')$.*

Proof. Let $\chi(x_0) = (\exists x_1) \ldots (\exists x_k) C_\chi(\vec{x}^T)$ with $\vec{x}^T = (x_0, x_1, \ldots, x_k)$. Let $\mathbb{Z} \models C_\chi(n_{(i)}^T)$ with $n_{(i)}^T = (n_{i,0}, \ldots, n_{i,k})$ and $i \in \{1, 2\}$ such that $n_{1,0} < n_{2,0}$.

We start by considering the positive literals of χ. By the above the positive literals of χ are equivalent in $\mathbb{Z}$ to an inhomogeneous linear system

$$A\vec{x} = b, \tag{I}$$

with $A \in \mathbb{Z}^{l \times (k+1)}$ and $b \in \mathbb{Z}^{l \times 1}$, where l is the number of positive literals of χ. Let us denote by $\mathcal{I}$ the set of solutions of (I) and by $\mathcal{H}$ the set of solutions of the homogeneous system. Then $n_{(1)}, n_{(2)} \in \mathcal{I}$ and therefore $h_0 \coloneqq n_{(1)} - n_{(2)} \in \mathcal{H}$. Hence $m \cdot h_0 + n_{(1)} \in \mathcal{I}$ for all $m \in \mathbb{N}$.

Now consider a negative literal $p(x_0, \ldots, x_k) \neq 0$ of χ, where p is a linear polynomial in the variables $x_0, \ldots, x_k$ with coefficients in $\mathbb{Z}$. Let $q(m) \coloneqq p((m \cdot h_0 + n_{(1)})^T)$, then q is a linear polynomial in one variable and moreover by the assumptions we have $q(0) = p(n_{(1)}^T) \neq 0$. Hence, there clearly is at most one $k \in \mathbb{Z}$ such that $q(k) = 0$.

Let $p_1(x_0, \ldots, x_k) \neq 0, \ldots, p_r(x_0, \ldots, x_k) \neq 0$ be all the negative literals of χ, then we let

$$m_0 = \max\left(\{0\} \cup \bigcup_{i=1}^{r} \{m + 1 \mid m \in \mathbb{N}, q_i(m) = 0\} \right).$$

Clearly, the natural number m_0 exists and we have $\mathbb{Z} \models C_\chi((m \cdot h_0 + n_{(1)})^T)$ for all $m \in \mathbb{N}$ with $m \geq m_0$. Since $n_{1,0} < n_{2,0}$ we are done. □

We summarize the results of this section in the following proposition.

Proposition 98. *Let $\varphi(x)$ be a p-free $\exists_1$ formula. There exists an $n \in \mathbb{N}$ such that if φ has n solutions in $\mathbb{N}$, then there exists an infinite strictly descending sequence of integers $(k_i)_{i\in\mathbb{N}}$ with $\mathbb{Z} \models \varphi(k_i)$ for all $i \in \mathbb{N}$.*

Proof. Let $\chi_1, \dots, \chi_k$ be p-free components such that $\vdash \varphi \leftrightarrow \bigvee_{i=1}^{k} \chi_i$. Let $n = k+1$ and assume that φ has n solutions in $\mathbb{N}$. Then by the pigeonhole principle there is a component χ_{i_0} with two solutions in $\mathbb{N}$ and therefore χ_{i_0} has two solutions in $\mathbb{Z}$. Finally, we apply Lemma 97 to χ_{i_0}. □

4.3. A non-standard model

In this section we construct a family of non-standard structures for the language L_{LA} and we make use of the results from Sections 4.1 and 4.2 in order to show that these structures are models of the theory

$$(B + \mathrm{B2} + \mathrm{B3}) + \exists_1(L_{\mathrm{LA}})\text{-IND}^{R-}.$$

Let us start by introducing some terminology about the models of this theory. Since already the theory $[B, \mathrm{Open}(L_{\mathrm{LA}})\text{-IND}^{R-}]$ proves B1 and $\mathcal{V}$, the models of $(B + \mathrm{B2} + \mathrm{B3}) + \exists_1(L_{\mathrm{LA}})\text{-IND}^{R-}$ are composed of a copy of the natural numbers—the standard elements—and copies of the integers, which we call the non-standard elements. The elements of the models we construct below are pairs of the form $n^{[m]} = (m, n) \in \mathbb{N} \times \mathbb{Z}$ such that $m = 0$ implies $n \in \mathbb{N}$. If $m = 0$, then the element is a standard element, otherwise it is non-standard and belongs to the m-th copy of the integers. We call m the type of the element and n the value of the element. We start by defining an operation that will allow us to relate the types of the elements.

Definition 99. The function $\uparrow\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ is given by

$$n \uparrow m \coloneqq \begin{cases} n & \text{if } n \neq 0 \\ m & \text{if } n = 0 \end{cases}.$$

Definition 100. Let $I \in \mathbb{N}$, then the L_{LA} structure M_I consist of pairs of the form $n^{[m]}$ with $n \in \mathbb{Z}$, $m \in \mathbb{N}$ and $m \leq I$ such that if $m = 0$ then $n \in \mathbb{N}$. Furthermore, we let M_I interpret the non-logical symbols as follows

$$0^{M_I} = 0^{[0]},$$
$$s^{M_I}(n^{[m]}) = (n+1)^{[m]}, \text{ for } n^{[m]} \in M,$$
$$p^{M_I}(n^{[m]}) = \begin{cases} (n \dot{-} 1)^{[0]} & \text{if } m = 0, \\ (n-1)^{[m]} & \text{otherwise.} \end{cases}$$
$$n_1^{[m_1]} +^{M_I} n_2^{[m_2]} = (n_1 + n_2)^{[m_1 \uparrow m_2]}.$$

The structure M_0 is isomorphic to the standard model $\mathbb{N}$. Since we are interested in constructing non-standard structures, we will consider mainly the structures M_I with $I \geq 1$.

Lemma 101. *Let $I \in \mathbb{N}$, then $M_I \models B + \mathrm{B1} + \mathrm{B3} + \mathcal{V}$.*

Proof. Routine. □

The structures M_I with $I \in \mathbb{N}$ and $I > 0$ have the crucial property that $\mathbb{Z}|_{\{s,p,+\}}$ can be embedded into the non-standard parts of M_I. Hence the solutions of 0-free $\exists_1$ formulas in $\mathbb{Z}$ can be transferred into the non-standard chains of M_I.

Lemma 102. *Let $m, I \in \mathbb{N}$ with $1 \leq m \leq I$, then the function $\iota_m : \mathbb{Z} \to M_I$, $n \mapsto n^{[m]}$ is an embedding of $\mathbb{Z}|_{\{s,p,+\}}$ into $M_I|_{\{s,p,+\}}$. In particular, if $\varphi(x)$ is a 0-free $\exists_1$ formula, then $\mathbb{Z} \models \varphi(n)$ implies $M_I \models \varphi(n^{[m]})$.*

Proof. It is routine to verify that ι_m is an embedding of $\mathbb{Z}|_{\{s,p,+\}}$ into $M_I|_{\{s,p,+\}}$. The second part of the claim follows from the well-known fact that embeddings preserve $\exists_1$ formulas (see for example [16, Theorem 2.4.1]) □

We can now show that the structures M_I satisfy unnested applications of the induction rule $\exists_1(L_{\mathrm{LA}})\text{-IND}^{R-}$.

Theorem 103. *Let $I \geq 1$ and T be a sound L_{LA} theory such that $M_I \models T$, then $M_I \models [T, \exists_1(L_{\mathrm{LA}})\text{-IND}^{R-}]$.*

Proof. Let $\varphi(x)$ be an $\exists_1$ formula and assume that φ is T-inductive. Since T is sound, we have $\mathbb{N} \models \varphi(x)$. Now consider an element $n^{[m]} \in M$. If $m = 0$, then $n \in \mathbb{N}$ and by the observation above $\mathbb{N} \models \varphi(\overline{n})$. By the $\exists_1$-completeness of B we have $B \vdash \varphi(\overline{n})$ and therefore $M_I \models \varphi(\overline{n})$. Since $M_I \models \overline{n} = n^{[0]}$ we obtain $M_I \models \varphi(n^{[m]})$.

Now assume that $m > 0$. By Proposition 95 there exists a 0-free p-free formula φ' and an $N \in \mathbb{N}$ such that $B + \mathrm{B1} + \mathcal{V} \vdash \varphi(s^N(x)) \leftrightarrow \varphi'(x)$. Hence we have $\mathbb{N} \models \varphi'$ and therefore by Proposition 98 there is an infinite strictly descending sequence of integers $(k_i)_{i\in\mathbb{N}}$ such that $\mathbb{Z} \models \varphi'(k_i)$ for $i \in \mathbb{N}$. By Lemma 102 we obtain $M_I \models \varphi'(k_i^{[m]})$ for $i \in \mathbb{N}$. Hence there exists $i_0 \in \mathbb{N}$ such that $k_{i_0} + N \leq n$ and $M_I \models \varphi'(k_{i_0}^{[m]})$, thus, $M_I \models \varphi((k_{i_0} + N)^{[m]})$. Since $M_I \models \varphi(x) \to \varphi(s(x))$, we obtain $M_I \models \varphi(k_{i_0} + N + k)^{[m]}$ for all $k \in \mathbb{N}$. In particular, we have $M_I \models \varphi(n^{[m]})$. □

By iterating the argument above we can show that the structures M_I even satisfy nested applications of $\exists_1(L_{\mathrm{LA}})$-IND^{R-}.

Corollary 104. *Let $I \geq 1$ and T be a sound L_{LA} theory such that $M_I \models T$, then $M_I \models T + \exists_1(L_{\mathrm{LA}})$-IND^{R-}.*

Proof. We proceed by induction on j to show $[T, \exists_1(L_{\mathrm{LA}})\text{-IND}^{R-}]_j$. For the base case we have $[T, \exists_1(L_{\mathrm{LA}})\text{-IND}^{R-}]_0 = T$, hence we are done. For the induction step assume that $M_I \models [T, \exists_1(L_{\mathrm{LA}})\text{-IND}^{R-}]_j$ and observe that $[T, \exists_1(L_{\mathrm{LA}})\text{-IND}^{R-}]_j$ is sound. Now obtain $M_I \models [T, \exists_1(L_{\mathrm{LA}})\text{-IND}^{R-}]_{j+1}$ by Theorem 103. □

Lemma 105. $M_1 \models \mathrm{B2}$.

Proof. Clearly it suffices to show that $b_1 \restriction b_2 = b_2 \restriction b_1$ for $b_1, b_2 \in \{0, 1\}$. The only interesting case is $b_1 \neq b_2$, that is, $0 \restriction 1 = 1 = 1 \restriction 0$. □

Corollary 106. $M_1 \models (B + \mathrm{B2} + \mathrm{B3}) + \exists_1(L_{\mathrm{LA}})$-IND^{R-}.

Proof. An immediate consequence of Lemma 105 and Corollary 104. □

Theorem 53 can now finally be obtained as an immediate consequence of Corollary 106.

Proof of Theorem 53. By Corollary 106 we can work with M_1. Now observe that $n \cdot k^{[1]} + \overline{(m-n)k} = (nk)^{[1]} + ((m-n)k)^{[0]} = (nk + (m-n)k)^{[1]} = (mk)^{[1]} = m \cdot k^{[1]}$, but $k^{[1]} \neq k^{[0]}$. □

Let us now consider whether some straightforward modifications of the background theory in Theorem 53 will improve the result. The following lemma shows that we do not strengthen the result of Theorem 53 by adding any $\forall_1$ consequence of $(B + \mathrm{B2} + \mathrm{B3}) + \exists_1(L_{\mathrm{LA}})$-IND^{R-} to the background theory.

Lemma 107. *Let L be a first-order language, T an L, $\forall_1$ theory and U a theory, then*

$$[T + U, \exists_1(L)\text{-IND}^{R-}]_n \equiv T + [U, \exists_1(L)\text{-IND}^{R-}]_n, \text{ for all } n \in \mathbb{N}.$$

Furthermore, $(T + U) + \exists_1(L)$-IND$^{R-} \equiv T + (U + \exists_1(L)$-IND$^{R-})$.

Proof. The first part is obtained by a straightforward induction on n and applying Lemma 44. The second part is an immediate consequence of the first part. □

The next natural question to ask is whether removing the formulas B2 and B3 in Theorem 53 would weaken the result. The following result shows that removing the axiom B2 would indeed weaken the result.

Lemma 108. $(B + \mathrm{B3}) + \exists_1(L_{\mathrm{LA}})$-IND$^{R-} \nvdash \mathrm{B2}$.

Proof. By Lemma 101 we have $M_2 \models B + \mathrm{B3}$. Since $0^{[1]} + 0^{[2]} = 0^{[1 \restriction 2]} = 0^{[1]} \neq 0^{[2]} = 0^{[2 \restriction 1]} = 0^{[2]} + 0^{[1]}$, we obtain $M_2 \models (B + \mathrm{B3}) + \exists_1(L_{\mathrm{LA}})$-IND$^{R-} \nvdash \mathrm{B2}$ by Corollary 104. □

We conjecture that removing B3 from Theorem 53 would weaken the result as well.

Conjecture 109. $(B + \mathrm{B2}) + \exists_1(L_{\mathrm{LA}})$-IND$^{R-} \nvdash \mathrm{B3}$.

We conclude this section by observing that the model-theoretic construction developed in this section does not yield a proof of Conjecture 58.

Lemma 110. *Let $I \geq 1$, then $M_I \not\models \exists_1(L_{\mathrm{LA}})\text{-IND}^-$.*

Proof. Let $\chi(x) = (\exists y_1)(\exists y_2)(\exists y_3)\theta(x, y_1, y_2, y_3)$ with

$$\theta(x, y_1, y_2, y_3) \coloneqq x + y_1 \neq x + y_2 \wedge x + (y_3 + y_1) = x + (y_3 + y_2).$$

We will show that $M_I \not\models I_x\chi(x)$. We first show that $M_I \models \chi(n^{[0]})$ for all $n \in \mathbb{N}$. For this it suffices to observe $n^{[0]} + 0^{[0]} = n^{[0]}$, $n^{[0]} + 0^{[1]} = n^{[1]}$ and $n^{[0]} + (0^{[1]} + 0^{[0]}) = n^{[0]} + 0^{[1]} = n^{[1]} = n^{[0]} + (0^{[1]} + 0^{[1]})$. Hence $M_I \models \theta(n^{[0]}, 0^{[0]}, 0^{[1]}, 0^{[1]})$. Now we will show that $M_I \not\models \chi(n^{[m]})$ for $n \in \mathbb{Z}$ and $m > 0$. Let $k_1^{[m_1]}$, $k_2^{[m_2]}$, $l^{[h]} \in M_I$ and assume that

$$n^{[m]} + k_1^{[m_1]} \neq n^{[m]} + k_2^{[m_2]}, \tag{*}$$

$$n^{[m]} + (l^{[h]} + k_1^{[m_1]}) = n^{[m]} + (l^{[h]} + k_2^{[m_2]}). \tag{†}$$

Since $m > 0$, we have $m \upharpoonright u = m$ for all $u \in \mathbb{Z}$. Hence by (*) we obtain $n + k_1 \neq n + k_2$. By (†) we obtain $n + l + k_1 = n + l + k_2$, thus $k_1 = k_2$. Therefore $n + k_1 = n + k_2$. Contradiction!

By the above we thus have $M_I \models \chi(0)$. Now let $n^{[m]} \in M_I$. If $m = 0$, then we have $M_I \models \chi((n+1)^{[0]})$ hence $M_I \models \chi(n^{[0]}) \to \chi((n+1)^{[0]})$. If $m > 0$, then we have $M_I \not\models \chi(n^{[m]})$ hence $M_I \models \chi(n^{[m]}) \to \chi((n+1)^{[m]})$. Thus $M_I \not\models I_x\chi(x)$. □

On the other hand it may be interesting to observe that already unnested applications of the parameter-free induction rule for $\exists_1$ formulas contain the parameter-free induction schema for quantifier-free formulas.

Lemma 111. *Let L be a language, then $[\varnothing, \exists_1(L)\text{-IND}^{R-}] \vdash \mathrm{Open}(L)\text{-IND}^-$.*

Proof. Let $\varphi(x)$ be a quantifier-free formula. Let $\psi(x, y)$ be given by

$$(\varphi(0) \wedge (\varphi(y) \to \varphi(s(y))) \to \varphi(x).$$

By shifting the existential quantifier inward, it is straightforward to see that $\vdash (\forall x)(\exists y)\psi(x, y) \leftrightarrow I_x\varphi$. Moreover, we clearly have $\vdash (\exists y)\psi(0, y)$. Now work in $\varnothing$ and assume $\psi(x, y_0)$. Assume furthermore $\varphi(0)$, $\varphi(y_0) \to \varphi(s(y_0))$ and $\varphi(x) \to \varphi(s(x))$. By the assumptions we obtain $\varphi(x)$ and moreover $\varphi(s(x))$. Hence, we have $(\exists y)\psi(s(x), y)$. Therefore, $\vdash (\exists y)\psi(x, y) \to (\exists y)\psi(s(x), y)$. Thus, $[\varnothing, \exists_1(L)\text{-IND}^{R-}] \vdash (\forall x)(\exists y)\psi(x, y)$. □

5. Conclusion

Clause set cycles are a formalism introduced by the authors of this article in [18] for the purpose of giving an upper bound on the strength on a family of AITP systems based on the extension of a saturation theorem prover by a cycle detection mechanism, such as the n-clause calculus [23,22]. In this article we have extended the analysis of clause set cycles that was begun in [18] by providing a logical characterization of refutation by a clause set cycle and concrete clause sets that are not refutable by a clause set cycle but that are refuted by induction for quantifier-free formulas.

In Section 3 we have identified several logical features of clause set cycles. Identifying these features has enabled us to give a characterization of the notion of refutation by a clause set cycle in terms of a logical theory. The characterization allows us to think of clause set cycles essentially as unnested applications of the parameter-free $\exists_1$ η induction rule. In the light of this logical characterization we were able to reduce the task of finding clause sets that are not refuted by a clause set cycle to an independence problem.

Based on this characterization we have shown two unprovability results for clause set cycles. The first result (Corollary 49) exploits the fact that refutations by a clause set cycle only make use of η-instances of the inductive lemmas. In particular, we have shown that even the full induction schema subject to the η-restriction does not prove some atoms that can already be obtained by an unnested application of the open parameter-free induction rule. This shows that the η-restriction is very limiting. However, our second unprovability result (Corollary 55) does not rely on the η-restriction and thus shows that AITP systems based on clause set cycles have more limitations. In Section 4 we have developed the underlying independence result (Theorem 53). This independence result shows us that the unprovability persists even when the induction rule is nested. We conjecture that this unprovability phenomenon is due to the absence of induction parameters and therefore also persists when the induction rule is replaced by the induction schema. This second unrefutability result shows that clause set cycles fail to capture induction arguments that involve very simple generalizations.

The results in this article together with the results in [18] explain much about the situation of AITP systems based on clause set cycles in the logical landscape. We have summarized the current results as well as some conjectures in Fig. 1. The figure depicts the refutational strength of various induction systems. The set of clause sets refuted by a system is described by an arc. The name of the system is inscribed near the top of the corresponding arc. The systems range over all first-order languages. The system CSC denotes refutation by a clause set cycle and NCC denotes the n-clause calculus as described in [18]. The points $\{\bullet_i \mid i = 1, 2, 3, 4, 5\}$ represent clause sets whose positions are confirmed by the results in this article and [18]. In particular $\bullet_1$ corresponds to the clause set that witnesses [18, Corollary 5.8], $\bullet_2$ corresponds to the clause set

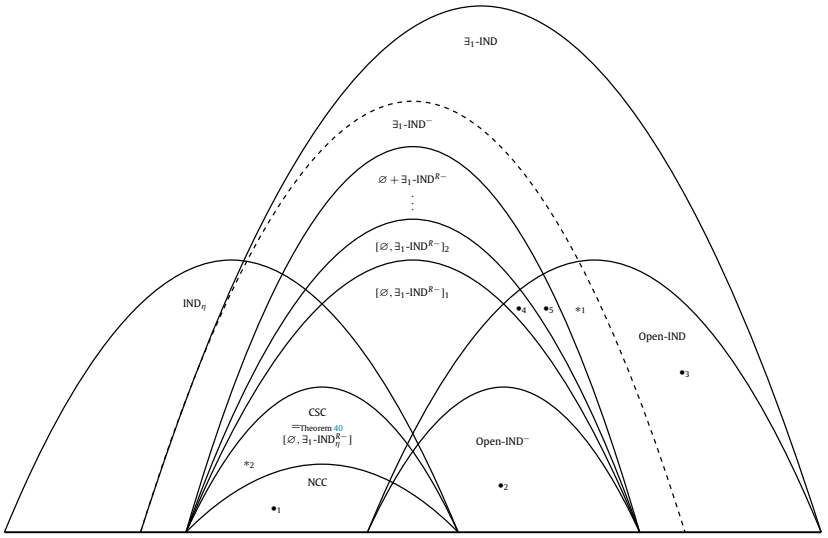

Fig. 1. Overview of the clausal refutational strength of various induction systems.

constructed in Section 3.3.1, and $\bullet_3$ corresponds to the clause sets constructed in Section 3.3.2, the points $\bullet_4$, $\bullet_5$ correspond to some the clause sets mentioned in Theorem 60. The inclusion of Open-IND$^-$ in $[\varnothing, \exists_1\text{-IND}^{R-}]$ is shown by Lemma 111.

The dashed arc corresponding to the system $\exists_1$-IND$^-$ is positioned according to Conjecture 58. The points $\{*_i \mid i = 1, 2\}$ of Fig. 1 represent clause sets that we conjecture to be at the respective positions. In particular, the point $*_1$ corresponds to the clause set mentioned in Conjecture 78. We would like to clarify the status of the point $*_1$ and the dashed arc corresponding to the system $\exists_1$-IND$^-$, as this would contribute to the understanding of the role of induction parameters and induction rules in automated inductive theorem proving.

Due to the recent advances in saturation-based theorem proving, the research on automated inductive theorem proving has recently increasingly focused on the integration of induction into saturation-based theorem provers [11,12,23,22,35,13,29,15]. We plan to carry out similar investigations for all these methods in order to develop a more global and unified view of induction in saturation-based theorem proving. In particular these investigations will give rise to the analysis of the interaction of the induction principle with various mechanisms of saturation-based provers such as Skolemization, splitting, term orderings, and redundancy criteria.

The point $*_2$ in Fig. 1 gives rise to a more general topic that is worth mentioning separately. On the one hand it is computationally expensive for AITP systems to carry out even a small number of inductions, and on the other hand the space of all possible induction formulas is very large. Hence AITP systems rely on heuristics to find induction formulas such as restricting the overall shape of the considered induction formulas and drawing syntactical material for induction from the formulas generated during the proof search. For example, the n-clause calculus as described in [23,22] only makes use of clause set cycles that appear as a subset of the clauses that are generated by the underlying saturation-based system. Such heuristics will not succeed in cases where a sufficiently non-analytic induction is required. Our technique for analyzing AITP systems as logical theories can deal with such heuristics only to a limited extent. For example, the notion of refutation by a clause set cycle completely ignores the fact that the n-clause calculus draws clause set cycles only from the generated clauses. Once the logical strength of most inductive theorem provers is known precisely enough it will likely be necessary to investigate the fine grained analyticity properties of the provers in order to get a better understanding of the consequences of restricting the degree of analyticity.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

We thank Emil Jeřábek for pointing out to us the similarity of clause set cycles with unnested induction rules. Moreover, we thank the anonymous reviewers whose feedback helped to improve the article significantly.

References

[1] Zofia Adamowicz, Parameter-free induction, the Matiyasevič theorem and BΣ_1, in: F.R. Drake, J.K. Truss (Eds.), Logic Colloquium '86, in: Studies in Logic and the Foundations of Mathematics, vol. 124, Elsevier, 1987, pp. 1–8.

[2] Michael Beeson, Mathematical induction in Otter-lambda, J. Autom. Reason. 36 (4) (2006) 311–344.
[3] Lev D. Beklemishev, Induction rules, reflection principles, and provably recursive functions, Ann. Pure Appl. Log. 85 (3) (1997) 193–242.
[4] Lev D. Beklemishev, Parameter free induction and reflection, in: Georg Gottlob, Alexander Leitsch, Daniele Mundici (Eds.), Computational Logic and Proof Theory, in: Lecture Notes in Computer Science, vol. 1289, Springer, 1997, pp. 103–113.
[5] Lev D. Beklemishev, Parameter free induction and provably total computable functions, Theor. Comput. Sci. 224 (1) (1999) 13–33.
[6] Lev D. Beklemishev, Reflection principles and provability algebras in formal arithmetic, Russ. Math. Surv. 60 (2) (apr 2005) 197–268.
[7] Siani Baker, Andrew Ireland, Alan Smaill, On the use of the constructive omega-rule within automated deduction, in: Andrei Voronkov (Ed.), Logic Programming and Automated Reasoning, in: Lecture Notes in Computer Science, vol. 624, Springer, 1992, pp. 214–225.
[8] Andrés Cordón-Franco, Alejandro Fernández-Margarit, Francisco Félix Lara Martín, A note on parameter free Π_1-induction and restricted exponentiation, Math. Log. Q. 57 (5) (2011) 444–455.
[9] Koen Claessen, Moa Johansson, Dan Rosén, Nicholas Smallbone, Automating inductive proofs using theory exploration, in: Maria Paola Bonacina (Ed.), Automated Deduction - CADE-24, in: Lecture Notes in Computer Science, vol. 7898, Springer, 2013, pp. 392–406.
[10] Koen Claessen, Moa Johansson, Dan Rosén, Nicholas Smallbone, TIP: tons of inductive problems, in: Manfred Kerber, Jacques Carette, Cezary Kaliszyk, Florian Rabe, Volker Sorge (Eds.), Intelligent Computer Mathematics, in: Lecture Notes in Computer Science, vol. 9150, Springer, 2015, pp. 333–337.
[11] Simon Cruanes, Extending Superposition with Integer Arithmetic, Structural Induction, and Beyond, PhD thesis École Polytechnique, Palaiseau, France, September 2015.
[12] Simon Cruanes, Superposition with structural induction, in: Clare Dixon, Marcelo Finger (Eds.), Frontiers of Combining Systems, in: Lecture Notes in Computer Science, vol. 10483, Springer, 2017, pp. 172–188.
[13] Mnacho Echenim, Nicolas Peltier, Combining induction and saturation-based theorem proving, J. Autom. Reason. 64 (2) (2020) 253–294.
[14] Gerhard Gentzen, Zusammenfassung von mehreren vollständigen Induktionen zu einer einzigen, Arch. Math. Log. Grundl.forsch. 2 (1) (1954) 1–3.
[15] Márton Hajdú, Petra Hozzová, Laura Kovács, Johannes Schoisswohl, Andrei Voronkov, Induction with generalization in superposition reasoning, in: Christoph Benzmüller, Bruce R. Miller (Eds.), Intelligent Computer Mathematics, in: Lecture Notes in Computer Science, vol. 12236, Springer, 2020, pp. 123–137.
[16] Wilfrid Hodges, A Shorter Model Theory, Cambridge University Press, 1997.
[17] Petr Hájek, Pavel Pudlák, Metamathematics of First-Order Arithmetic, Perspectives in Mathematical Logic, Springer, 1993.
[18] Stefan Hetzl, Jannik Vierling, Clause set cycles and induction, Log. Methods Comput. Sci. 16 (4) (November 2020) 11.
[19] Stefan Hetzl, Tin Lok Wong, Some observations on the logical foundations of inductive theorem proving, Log. Methods Comput. Sci. 13 (4) (April 2018) 10.
[20] Emil Jeřábek, Induction rules in bounded arithmetic, Arch. Math. Log. 59 (3) (5 2020) 461–501.
[21] Moa Johansson, Dan Rosén, Nicholas Smallbone, Koen Claessen, Hipster: integrating theory exploration in a proof assistant, in: Stephen M. Watt, James H. Davenport, Alan P. Sexton, Petr Sojka, Josef Urban (Eds.), Intelligent Computer Mathematics, in: Lecture Notes in Computer Science, vol. 8543, Springer, 2014, pp. 108–122.
[22] Abdelkader Kersani, Preuves par induction dans le calcul de superposition, PhD thesis, Université de Grenoble, October 2014.
[23] Abdelkader Kersani, Nicolas Peltier, Combining superposition and induction: a practical realization, in: Pascal Fontaine, Christophe Ringeissen, Renate A. Schmidt (Eds.), Frontiers of Combining Systems, in: Lecture Notes in Computer Science, vol. 8152, Springer, 2013, pp. 7–22.
[24] Richard Kaye, Jeff Paris, Costas Dimitracopoulos, On parameter free induction schemas, J. Symb. Log. 53 (4) (1988) 1082–1097.
[25] Yutaka Nagashima, LiFtEr: language to encode induction heuristics for Isabelle/HOL, in: Anthony Widjaja Lin (Ed.), Programming Languages and Systems, in: Lecture Notes in Computer Science, vol. 11893, Springer, 2019, pp. 266–287.
[26] Charles Parsons, On **n**-quantifier induction, J. Symb. Log. 37 (3) (1972) 466–482.
[27] Uday S. Reddy, Term rewriting induction, in: Mark E. Stickel (Ed.), 10th International Conference on Automated Deduction, in: Lecture Notes in Computer Science, vol. 449, Springer, 1990, pp. 162–177.
[28] H.E. Rose, Subrecursion: Functions and Hierarchies, Oxford University Press, 1984.
[29] Giles Reger, Andrei Voronkov, Induction in saturation-based proof search, in: Pascal Fontaine (Ed.), Automated Deduction - CADE 27, in: Lecture Notes in Computer Science, vol. 11716, Springer, 2019, pp. 477–494.
[30] John Cedric Shepherdson, Non-standard models for fragments of number theory, in: J.W. Addison, Leon Henkin, Alfred Tarski (Eds.), The Theory of Models, in: Studies in Logic and the Foundations of Mathematics, North-Holland, 1963, pp. 342–358.
[31] Joseph Robert Shoenfield, Open sentences and the induction axiom, J. Symb. Log. 23 (1) (1958) 7–12.
[32] Wilfried Sieg, Herbrand analyses, Arch. Math. Log. 30 (5–6) (1991) 409–441.
[33] Irene Lobo Valbuena, Moa Johansson, Conditional lemma discovery and recursion induction in Hipster, Electron. Commun. EASST 72 (2015).
[34] Andrei Voronkov, AVATAR: the architecture for first-order theorem provers, in: Armin Biere, Roderick Bloem (Eds.), Computer Aided Verification, in: Lecture Notes in Computer Science, vol. 8559, Springer, 2014, pp. 696–710.
[35] Daniel Wand, Superposition: Types and Induction, PhD thesis, Saarland University, Saarbrücken, Germany, 2017.