Revisiting the conservativity of fixpoints over intuitionistic arithmetic

Mattias Granberg Olsson[1] · Graham E. Leigh[1]

Abstract

This paper presents a novel proof of the conservativity of the intuitionistic theory of strictly positive fixpoints, $\widehat{\mathrm{ID}}_1^i$, over Heyting arithmetic (HA), originally proved in full generality by Arai (Ann Pure Appl Log 162:807–815, 2011. https://doi.org/10.1016/j.apal.2011.03.002). The proof embeds $\widehat{\mathrm{ID}}_1^i$ into the corresponding theory over Beeson's logic of partial terms and then uses two consecutive interpretations, a realizability interpretation of this theory into the subtheory generated by almost negative fixpoints, and a direct interpretation into Heyting arithmetic with partial terms using a hierarchy of satisfaction predicates for almost negative formulae. It concludes by applying van den Berg and van Slooten's result (Indag Math 29:260–275, 2018. https://doi.org/10.1016/j.indag.2017.07.009) that Heyting arithmetic with partial terms plus the schema of self realizability for arithmetic formulae is conservative over HA.

Keywords Fixpoint theories · Conservativity · Realizability · Satisfaction predicates

Mathematics Subject Classification 03F50 · 03F25 · 03F30 · 03F55

✉ Mattias Granberg Olsson
mattias.granberg.olsson@gu.se

Graham E. Leigh
graham.leigh@gu.se

[1] Department of Philosophy, Linguistics and Theory of Science, University of Gothenburg, Box 200, 405 30 Göteborg, Sweden

1 Introduction

An occurrence of a symbol in a formula φ is *strictly positive* if the occurrence is not inside an antecedent of an implication in φ. Given a parameter predicate P we can thus form all formulae which have only strictly positive occurrences of P, referred to as the *strictly positive operator forms*. Given such an operator form $\Phi(P; x)$ with free variable x and writing $\Phi(\varphi; x)$ for the result of substituting φ for P in Φ, a predicate I_Φ satisfying

$$\forall x (I_\Phi(x) \leftrightarrow \Phi(I_\Phi; x)) \tag{1}$$

is then a *fixpoint* of the operator form Φ. The theory $\widehat{\mathrm{ID}}_1$ is Peano arithmetic (PA) extended with a new symbol I_Φ and axiom (1) for all strictly positive operator forms Φ, and $\widehat{\mathrm{ID}}_1^i$ is the corresponding extension of Heyting arithmetic (HA). Inductively, $\widehat{\mathrm{ID}}_{n+1}$ and $\widehat{\mathrm{ID}}_{n+1}^i$ are the corresponding extensions of $\widehat{\mathrm{ID}}_n$ and $\widehat{\mathrm{ID}}_n^i$ respectively. $\widehat{\mathrm{ID}}_1$ is not conservative over PA, since already $\widehat{\mathrm{ID}}_1(\Pi_2)$, in which (1) holds only for strictly positive Π_2 operator forms Φ, proves the consistency of PA. On the contrary, $\widehat{\mathrm{ID}}_1^i$ is known to be conservative over HA.

In the 1997 paper [1], Wilfried Buchholz proved that the theory of fixpoints for *strongly* positive operator forms, that is where (1) holds for operator forms Φ which contain no implications whatsoever, is conservative over HA for almost negative sentences. The result was soon improved to full conservativity by Toshiyasu Arai in [2]. A few years later (2002), Christian Rüede and Thomas Strahm in [3] made an improvement in another direction, by showing that the theory $\widehat{\mathrm{ID}}_\alpha^i$, where α ($< \Gamma_0$) refers to ordinal iterations of the fixpoint construction, is conservative over the theory ACA_α^{-i} (intuitionistic theory of iterated arithmetical comprehension without set parameters) for negative and Π_2^0 sentences. Their argument makes a realizability interpretation of $\widehat{\mathrm{ID}}_\alpha^i$ in an appropriate fragment, which is subsequently interpreted in the classical theory ACA_α^- via partial truth predicates; finally ACA_α^- is conservative over ACA_α^{-i} for negative and Π_2^0 sentences. As a corollary, $\widehat{\mathrm{ID}}_n^i$ is conservative over HA for such sentences.

A few years after this series of partial results (2011), Arai in [4] finally showed, using cut-elimination of an infinitary derivation system formalised in HA, that the theory $\widehat{\mathrm{ID}}_1^i$ is fully conservative over HA (and the stronger result that all $\widehat{\mathrm{ID}}_n^i$ are conservative over HA).

The aim of this paper is to reprove this important result by a new method. For clarity we first outline our argument for conservation of almost negative sentences only. This closely resembles that of Rüede and Strahm outlined above in that it combines two natural translations: a realizability interpretation of $\widehat{\mathrm{ID}}_1^i$ in the subtheory with the fixpoint axiom (1) restricted to almost negative operator forms, and a direct interpretation of this subtheory in Heyting arithmetic. As in [3], the first step uses standard Kleene-style realizability in the manner of Buchholz' [5, §6]. The main difference lies in the second reduction, which interprets fixpoint predicates for almost negative operator forms by partial satisfaction predicates definable in Heyting arithmetic and generalises the conservativity of fixpoints for strictly positive Π_1 operator forms over

Peano arithmetic. In this way, the detour through classical logic used in [3] can be avoided and conservativity of $\widehat{\mathsf{ID}}_1^{\mathsf{i}}$ over HA obtained also for almost negative sentences outside Π_2^0, since, combining the two interpretations, if φ is an arithmetic theorem of $\widehat{\mathsf{ID}}_1^{\mathsf{i}}$ we obtain that φ is realizable in HA and thus HA $\vdash \varphi$ if φ is almost negative.

Full conservation can be obtained by the same argument if the theories in question are formulated over Beeson's logic of partial terms (see [6]) rather than intuitionistic predicate logic. Let HAP and $\widehat{\mathsf{ID}}_1^{\mathsf{i}}\mathsf{P}$ be Heyting arithmetic with partial terms and its extension to fixpoints for strictly positive operator forms respectively. We show that every arithmetic theorem of $\widehat{\mathsf{ID}}_1^{\mathsf{i}}$ is realized in HAP via an analogous combination of interpretations. This involves a realizability interpretation of $\widehat{\mathsf{ID}}_1^{\mathsf{i}}\mathsf{P}$ in the subtheory of almost negative fixpoints, which we then show is a conservative extension of HAP. This conservativity is likewise witnessed by the definability of partial satisfaction predicates for almost negative formulae involving partial terms. The final step of concluding HA $\vdash \varphi$ from the realizability of φ in HAP is a consequence of a result due to Benno van den Berg and Lotte van Slooten in [7] that for this realizability interpretation, Heyting arithmetic is conservatively extended by HAP$+\varphi \leftrightarrow \exists x(x\,\underline{\mathsf{r}}\,\varphi)$ for arithmetic φ, where $x\,\underline{\mathsf{r}}\,\varphi$ expresses 'x realizes φ.'

We will start by fixing basic notation and terminology in Sect. 2. Heyting arithmetic with partial terms and the basic realizability interpretation we utilise is introduced in Sect. 3. The main results from [7] which we require are also rehearsed in that section. Section 4 concerns properties of the almost negative formulae. We present a hierarchy of formulae $(\Lambda_n)_n$ based on nesting depth of implications which exhausts the almost negative formulae and present for each n a Λ_n-formula that provably in HAP is a satisfaction predicate for Λ_n-formulae. Section 5 overviews three intuitionistic fixpoint theories of import: the theory $\widehat{\mathsf{ID}}_1^{\mathsf{i}}$ of strictly positive fixpoints over Heyting arithmetic, the counterpart theory over the logic partial terms, $\widehat{\mathsf{ID}}_1^{\mathsf{i}}\mathsf{P}$, and its subtheory of fixpoints for almost negative operator forms, $\widehat{\mathsf{ID}}_1^{\mathsf{i}}\mathsf{P}(\Lambda)$, as well as interpretability results among them. The article concludes with proving the main result and a discussion of the methods and potential extensions, in Sect. 6.

This paper constitutes part of the first authors thesis, and the details of the omitted proofs will be presented in the forthcoming dissertation.

2 Preliminaries

Our base languages will be $\mathcal{L}_{\mathsf{HA}}$, $\mathcal{L}_{\mathsf{PRA}}$ and $\mathcal{L}_{\mathsf{HAP}}$, all of which will contain the quantifiers $\forall$ and $\exists$, the connectives $\wedge$, $\vee$ and $\rightarrow$, the propositional constant $\bot$ and the equality relation $=$ as logical symbols. $\mathcal{L}_{\mathsf{HA}}$, the language of Heyting arithmetic, in addition contains the constant symbol 0 and the function symbols S, $+$ and $\times$. Numerals in $\mathcal{L}_{\mathsf{HA}}$ will be constructed from 0 and S in the usual way ($\overline{0} = 0$, $\overline{n+1} = \mathsf{S}(\overline{n})$); the same holds for all languages we will consider. $\mathcal{L}_{\mathsf{PRA}}$, the language of primitive recursive arithmetic, contains the symbols of $\mathcal{L}_{\mathsf{HA}}$ as well as a function symbol for each primitive recursive function (including fresh symbols for addition and multiplication). The final language, $\mathcal{L}_{\mathsf{HAP}}$, of Heyting arithmetic with partial terms extends $\mathcal{L}_{\mathsf{HA}}$ by constant symbols $\mathbf{k}$, $\mathbf{s}$, $\mathbf{p}_{\mathsf{l}}$, $\mathbf{p}_{\mathsf{r}}$, $\mathbf{p}$, $\mathbf{succ}$, $\mathbf{r}$ and the binary function symbol $\cdot$. Since $\cdot$ will

occur more frequently than $\times$ we will often abbreviate it by juxtaposition and let $\times$ be written out. HA (PA) is the intuitionistic (resp. classical) theory in $\mathcal{L}_{\mathrm{HA}}$ axiomatised by basic axioms for the successor S, defining equations for $+$ and $\times$, and induction for $\mathcal{L}_{\mathrm{HA}}$. Heyting arithmetic with partial terms, HAP, is the theory in $\mathcal{L}_{\mathrm{HAP}}$ presented in [7] which we will return to in Sect. 3. $\mathrm{HA}(\mathcal{L})$ and $\mathrm{HAP}(\mathcal{L})$ will be HA and HAP with induction extended (or restricted) to $\mathcal{L}$, but with no additional axioms concerning the non-arithmetical symbols. Finally, $\mathrm{HA}_{\mathcal{L}_{\mathrm{PRA}}}$ is Heyting arithmetic axiomatised in $\mathcal{L}_{\mathrm{PRA}}$ with defining equations for all function symbols as axioms, essentially as in, for example, Troelstra and van Dalen's [8]. While there is no real difference between HA and $\mathrm{HA}_{\mathcal{L}_{\mathrm{PRA}}}$ (they are mutually interpretable), the small number of symbols and the fact that $\mathcal{L}_{\mathrm{HA}} \subseteq \mathcal{L}_{\mathrm{HAP}}$ will be advantageous from our perspective; at the same time, several formula-classes are less expressive in HA than in $\mathrm{HA}_{\mathcal{L}_{\mathrm{PRA}}}$, which makes $\mathrm{HA}_{\mathcal{L}_{\mathrm{PRA}}}$ more desirable to work with in some situations.

We use an infinite sequence of variables, $\mathsf{v}_0, \mathsf{v}_1, \ldots$; metavariables for these will be denoted by $x, y, z, u, v, w, \ldots$ etc.. Terms are constructed from variables, constants and function symbols in the usual way; for disambiguation we will consider a variable and the term made up of only that variable as distinct syntactic objects, in particular when it comes to coding. We use $\tau, \sigma, \rho, \ldots$ as metavariables for terms. We denote tuples of terms (or variables) with $\vec{\tau}$, or $\vec{\tau}^n$ if we want to emphasise the tuple has length n. Symbols $\Phi, \Psi, \phi, \varphi, \psi, \theta, \vartheta, \ldots$ denotes formulae, where uppercase is used to emphasise that the formula is an operator form, in that it contains a distinguished predicate symbol (denoted by P, Q or R). Formulae are identified up to α-equivalence, and we define the Gödel code of a formula to be the (numerically) least code of any of the α-equivalent formulae with no *nested bindings*, i.e. two nested quantifiers binding the same variable. Regarding substitution: $\sigma[\vec{u}/\vec{\tau}]$, where σ is a term, $\vec{u}$ is a finite sequence of distinct variables and $\vec{\tau}$ is a sequence of terms of the same length, denotes simultaneous substitution of τ_i for u_i in σ for all indices i of the sequences. Similarly $\varphi(\vec{u}/\vec{\tau})$, where φ is a formula and $\vec{u}$ and $\vec{\tau}$ are as above, denotes simultaneous substitution of τ_i for the *free* occurrences of u_i in φ for all indices i of the sequences *after renaming bound variables* in φ to avoid conflicts. Since we identify α-equivalent formulae this is permissible. The simpler expression $\varphi(\vec{\tau})$ means $\varphi(\vec{\mathsf{v}}_{<n}/\vec{\tau})$, where $\vec{\mathsf{v}}_{<n}$ is $\langle \mathsf{v}_0, \ldots, \mathsf{v}_{n-1} \rangle$, with n the length of $\vec{\tau}$. When introducing a formula with a formulation of the kind "Let $\varphi(\vec{x})$ be a formula," it is tacitly understood that $\vec{x}^n$ is without repetitions and no variables other than $\mathsf{v}_0, \ldots, \mathsf{v}_{n-1}$ occurs free in φ, that is all free occurrences of $\vec{x}$ in $\varphi(\vec{x})$ are obtained by the substitution; $\varphi(\vec{\tau})$ is subsequently the same as $\varphi(\vec{x})(\vec{x}/\vec{\tau})$. We use the corresponding notation for substitution of formulae for relations: $\Phi(R/\lambda\vec{x}.\vartheta)$, where the length of $\vec{x}$ is the arity of R, denotes simultaneous substitution in the formula Φ of the formula ϑ for the relation symbol R, where the ith argument of R is substituted for the free occurrences of x_i in ϑ, after renaming bound variables in Φ and ϑ.[1] Note, we do not require that x_i is a free variable in ϑ or that all free variables of ϑ are among $\vec{x}$.

---

[1] It is of course important that no variable occurring as an argument to R is bound in ϑ.

If we omit the '$\lambda \vec{x}$' we mean that $\vec{x}$ is $\vec{v}_{<n}$ and if ϑ consists of a single n-ary relation symbol Q we write $\Phi(R/Q)$ for $\Phi(R/\lambda \vec{v}_{<n}.Q(\vec{v}_{<n}))$, where we might also omit R if it is clear from the context (as in Sect. 5). Finally, when expressing both relation and term substitution, denoted $\Phi(\lambda \vec{x}.\vartheta; \vec{\tau})$, term substitution takes precedence, namely $\Phi(\lambda \vec{x}.\vartheta; \vec{\tau}) = (\Phi(\vec{\tau}))(\lambda \vec{x}.\vartheta)$.[2] We will abuse notation and write $\chi \in \mathcal{L}$ for any symbol, term or formula χ of the language $\mathcal{L}$. The Gödel code of an expressions η (in the language under consideration) will be denoted by $\ulcorner \eta \urcorner$. When using numerals of Gödel codes we suppress the bar, so that in e.g. $\mathrm{HA} \vdash \ulcorner \eta \urcorner \neq 0$, $\ulcorner \eta \urcorner$ refers to the numeral of the number $\ulcorner \eta \urcorner$.

We finally turn to notations for syntactic formula-classes. Λ will denote the class of almost negative formulae; a formula is *almost negative* if it contains no disjunctions and existential quantifiers only occur immediately in front of term-equations. Note the unfortunate clash of terminology that some strictly positive operator forms (meaning the distinguished predicate P does not occur in the antecedent of an implication) will at the same time be almost negative formulae. Both of these concepts are standard in their fields, so there is little point in trying to avoid or change them here. *Negative* formulae are the almost negative formulae which contains no existential quantifiers whatsoever. When concerned with a syntactic formula-class Γ such as Λ or Σ_n (or Λ_n introduced in Sect. 4) we will use the convention that Γ refers to the set of formulae belonging to this class in the language under consideration, while $\Gamma(\mathrm{HA})$ refers to the set of formulae which are HA-equivalent to formulae from Γ in this language; the same principle applies to other theories like HAP and PA. For example $\Sigma_1(\mathrm{PA})$ is the set of formulae in $\mathcal{L}_{\mathrm{HA}}$ which are PA-equivalent to Σ_1-formulae. Some caution must be used here since $\mathcal{L}_{\mathrm{HA}}$ does not contain the symbol $<$; Δ_0-formulae (and subsequently the rest of the arithmetical hierarchy) are thus defined via the defined inequality: $x < y$ is $\exists z(x + \mathsf{S}(z) = y)$. If HA' is Heyting arithmetic in $\mathcal{L}_{\mathrm{HA}} + <$, this yields HA' as a definitional extension of HA, with the arithmetical hierarchy in HA the image of the one in HA'.

3 LPT and HAP

For completeness and convenience we will begin by briefly describing the system LPT of Beeson's logic of partial terms (see [6]), followed by HAP, Heyting arithmetic in partial terms. Both of these presentations will be essentially as those given in [7]. Most proofs will be omitted; we refer the reader to the references for details.

Definition 3.1 (LPT) Logic of Partial Terms, LPT, is a Hilbert-style deductive system in a language containing at least equality with the following logical rules and axiom schemata, where $\tau \downarrow$ abbreviates $\tau = \tau$ and expresses that τ *denotes*:

---

[2] Note that all choices involved can be made in an unambiguous (primitive recursive) way, which we will assume without specifying one.

LPT1. $\phi \to \phi$
LPT2. $\phi, \phi \to \psi \Rightarrow \psi$
LPT3. $\phi \to \psi, \psi \to \chi \Rightarrow \phi \to \chi$
LPT4. $\phi \wedge \psi \to \phi, \phi \wedge \psi \to \psi$
LPT5. $\phi \to \psi, \phi \to \chi \Rightarrow \phi \to \psi \wedge \chi$
LPT6. $\phi \to \phi \vee \psi, \psi \to \phi \vee \psi$
LPT7. $\phi \to \chi, \psi \to \chi \Rightarrow \phi \vee \psi \to \chi$
LPT8. $\phi \wedge \psi \to \chi \Rightarrow \phi \to (\psi \to \chi)$
LPT9. $\phi \to (\psi \to \chi) \Rightarrow \phi \wedge \psi \to \chi$
LPT10. $\bot \to \phi$
LPT11. $\phi \to \psi \Rightarrow \phi \to \forall x \psi$ for $x \notin \mathrm{FV}(\phi)$
LPT12. $\forall x \phi \wedge \tau{\downarrow} \to \phi(x/\tau)$
LPT13. $\phi \to \psi \Rightarrow \exists x \phi \to \psi$ for $x \notin \mathrm{FV}(\psi)$
LPT14. $\phi(x/\tau) \wedge \tau{\downarrow} \to \exists x \phi$
LPT15. $\forall x (x = x)$
LPT16. $\forall x y (x = y \to y = x)$
LPT17. $\forall x y z (x = y \wedge y = z \to x = z)$
LPT18. $\forall \vec{x} \vec{y} (\vec{x} = \vec{y} \wedge \mathsf{f}(\vec{x}){\downarrow} \to \mathsf{f}(\vec{x}) = \mathsf{f}(\vec{y}))$
LPT19. $\forall \vec{x} \vec{y} (R(\vec{x}) \wedge \vec{x} = \vec{y} \to R(\vec{y}))$
LPT20. $\mathsf{c}{\downarrow}$
LPT21. $\mathsf{f}(\vec{\tau}){\downarrow} \to \tau_i{\downarrow}$
LPT22. $R(\vec{\tau}) \to \tau_i{\downarrow}$
LPT23. $x{\downarrow}$

Rules LPT11 and LPT13 are called (universal and existential) generalisation, axioms LPT12 and LPT14 (universal and existential) instantiation and LPT20, LPT21, LPT22 and LPT23 are the strictness axioms. An instance of LPT22 is $\tau = \sigma \to \tau{\downarrow} \wedge \sigma{\downarrow}$. Axiom LPT23 is not explicitly included in the axiomatisation in [7] but is consistent with it, being derivable from the the axioms of HAP. Since the axiom is present in Beeson's axiomatisation in [6] and the axiomatisation of Troelstra's [9], as well as [8] (there called E^+-logic), we include it for simplicity. This also makes sense for α-equivalence, which is built into this system by construction; given LPT23 one can also derive it from (more carefully formulated) LPT12 and LPT14.

Definition 3.2 If Γ is a set of formulae and ϕ a formula then $\Gamma \overset{\mathscr{D}}{\vdash} \phi$ means $\mathscr{D}$ is an LPT-derivation (sequence of formulae) of ϕ using Γ as additional axioms. We write $\Gamma \vdash \phi$ to express that $\Gamma \overset{\mathscr{D}}{\vdash} \phi$ for some LPT-derivation $\mathscr{D}$. If Γ is empty we write $\overset{\mathscr{D}}{\vdash} \phi$.

The notion of derivation introduced above satisfies the deduction theorem. We can also define weak equality and obtain:

Lemma 3.3 ("Leibniz Lemma" of [7]) *Defining $\sigma \simeq \tau$ as $\sigma{\downarrow} \vee \tau{\downarrow} \to \sigma = \tau$ we have $\vdash \vec{\sigma} \simeq \vec{\tau} \to \mathsf{f}(\vec{\sigma}) \simeq \mathsf{f}(\vec{\tau})$ and $\vdash \sigma \simeq \tau \wedge \phi(x/\tau) \to \phi(x/\sigma)$ for all function symbols f and all formulae ϕ of the language.*

Definition 3.4 (HAP) *Heyting arithmetic with partial terms*, HAP, is an LPT-theory in the language $\mathcal{L}_{\mathrm{HAP}}$. It is axiomatised by the following formulae for all variables x, y and z and formulae $\varphi \in \mathcal{L}_{\mathrm{HAP}}$. Recall, juxtaposition of terms abbreviates use of $\cdot$ (read as application).

HAP1. $\mathsf{S}(x)\downarrow, (x + y)\downarrow, (x \times y)\downarrow$
HAP2. $\mathsf{S}(x) = \mathsf{S}(y) \rightarrow x = y, 0 \neq \mathsf{S}(x)$
HAP3. $x + 0 = x, x + \mathsf{S}(y) = \mathsf{S}(x + y)$
HAP4. $x \times 0 = 0, x \times \mathsf{S}(y) = (x \times y) + x$
HAP5. $\mathbf{k}xy = x, (\mathbf{s}xy)\downarrow, \mathbf{s}xyz \simeq xz(yz)$
HAP6. $\mathbf{p}_\mathbf{l}x\downarrow, \mathbf{p}_\mathbf{r}x\downarrow, \mathbf{p}_\mathbf{l}(\mathbf{p}xy) = x, \mathbf{p}_\mathbf{r}(\mathbf{p}xy) = y, \mathbf{p}(\mathbf{p}_\mathbf{l}x)(\mathbf{p}_\mathbf{r}x) = x$
HAP7. $\mathbf{succ}x = \mathsf{S}(x), \mathbf{r}xy0 = x, \mathbf{r}xy\mathsf{S}(z) \simeq yz(\mathbf{r}xyz)$
HAP8. $\varphi(x/0) \rightarrow (\forall x(\varphi \rightarrow \varphi(x/\mathsf{S}(x))) \rightarrow \forall x\varphi)$

Terms and formulae from the fragment $0, \mathsf{S}, +, \times$ will be called *arithmetical*. A *combinator* (term) is a closed term without $\mathsf{S}, +$ or $\times$, making 0 the only arithmetical combinator; the other combinator constants $\mathbf{k}, \mathbf{s}, \mathbf{p}, \mathbf{p}_\mathbf{l}, \mathbf{p}_\mathbf{r}, \mathbf{succ}$ and $\mathbf{r}$ are the *pure* combinators. A combinator τ is *unary* (*binary* etc.) *total* if $\mathrm{HAP} \vdash \tau x\downarrow$ ($\mathrm{HAP} \vdash (\tau x)y\downarrow$ etc.). If $\mathcal{L}$ is a language extending $\mathcal{L}_{\mathrm{HAP}}$ we will extend HAP to an $\mathcal{L}$-theory $\mathrm{HAP}(\mathcal{L})$ by letting the induction schema HAP8 apply to all formulae of $\mathcal{L}$.

Remark 3.5 van den Berg and van Slooten's [7] used strong equality in the third part of axiom HAP7. Private communication with Benno van den Berg has revealed that this was not intended and that the results of [7] do not depend on it. We therefore use the version with weak equality, which is in line with Beeson's [6] and Troelstra and van Dalen's [11].

Not surprisingly HA directly embeds into HAP. The next lemma summarises the key observations for this result.

Lemma 3.6 *Let $\mathcal{L}_1 \subseteq \mathcal{L}_2$ be arbitrary first-order languages and Γ_1, Γ_2 sets of formulae in the respective language. Let $\vdash_\mathrm{I}$ denote derivability in ordinary intuitionistic logic, as captured by the Hilbert-style system* LPT *plus the extra logical axiom $\tau\downarrow$ for all terms τ. Suppose $\Gamma_2 \vdash \sigma\downarrow$ for all terms σ of $\mathcal{L}_1$. If $\Gamma_2 \vdash \gamma$ for all $\gamma \in \Gamma_1$ and $\Gamma_1 \vdash_\mathrm{I} \varphi$ then $\Gamma_2 \vdash \varphi$, for all $\varphi \in \mathcal{L}_1$.*

Proof Induction on derivations $\Gamma_1 \vdash_\mathrm{I} \varphi$. □

Corollary 3.7 HAP *proves all theorems of* HA.

3.1 Combinatory algebra in HAP

This subsection aims to describe how the basic facts of combinatory algebra are derivable in HAP. In this, we follow [6, VI.2.]. The basic idea is to regard every object of the theory both as a number and as a partial function, where $\cdot$ denotes function application to an argument. It is in line with this reading we will write $\tau\vec{\sigma}$ and $\tau \cdot \vec{\sigma}$ for $\tau \cdot \sigma_0 \cdot \cdots \cdot \sigma_n$, which in turn is a shorthand for $((\cdots(\tau \cdot \sigma_0) \cdot \cdots) \cdot \sigma_n)$; thus association of $\cdot$ is to the left.

It is well known that combinatory algebra is Turing-complete, and thus defines the same class of functions as the recursive functions or the λ-calculus. Adapted to present purposes we observe that we can define a denoting 'λ-term' for any partial term in the language, such that if applied to an argument they are weakly equal.

Proposition 3.8 (λ-terms, Proposition 2.3 of [7]) *For each term τ and variable x in $\mathcal{L}_{HAP}$, there is a term $\lambda x.\tau$ in the fragment without S, $+$ and $\times$ such that $FV(\lambda x.\tau) = FV(\tau)\backslash\{x\}$, $HAP \vdash \lambda x.\tau\downarrow$ and $HAP \vdash \sigma\downarrow \rightarrow (\lambda x.\tau) \cdot \sigma \simeq \tau[x/\sigma]$.*

As stated above we will call terms constructed in accordance with this lemma *λ-terms*. At first sight they might seem rather redundant, but note that a λ-term *always* denotes, as opposed to the term it was constructed from (though the application of a λ-term to some argument need not denote, of course). Note that this also means that τ is *not* a subterm of $\lambda x.\tau$ in general. Another consequence of the construction is that λ-terms do not contain any arithmetical function symbols, but only $\cdot$. We will use the usual notational conventions for stacking λs, that is $\lambda x.\lambda y.\tau$ will be written $\lambda xy.\tau$ etc.

Proposition 3.9 (Primitive recursion in HAP) *For each function symbol of $\mathcal{L}_{PRA}$ defined as a primitive recursive function there is a combinator of $\mathcal{L}_{HAP}$ such that HAP proves its defining equations. More precisely there is an interpretation τ of these function symbols of $\mathcal{L}_{PRA}$ into the combinator terms of $\mathcal{L}_{HAP}$ such that for all such f, g, and $h_{(j)}$ with arities n, $k+2$ and k*

$$HAP \vdash \forall x\ \tau_Z x = 0,$$
$$HAP \vdash \forall x\ \tau_S x = S(x),$$
$$HAP \vdash \forall \vec{x}\ \tau_{\pi_i^k}\vec{x} = x_i,$$
$$HAP \vdash \forall \vec{x}\ \tau_{\circ_n^k(f,h_0,\dots,h_{n-1})}\vec{x} = \tau_f(\tau_{h_0}\vec{x})\cdots(\tau_{h_{n-1}}\vec{x}),$$
$$HAP \vdash \forall \vec{x}\ \tau_{rec^k(g,h)}\vec{x}\,0 = \tau_h\vec{x},$$
$$HAP \vdash \forall \vec{x}y\ \tau_{rec^k(g,h)}\vec{x}(S(y)) = \tau_g\vec{x}y(\tau_{rec^k(g,h)}\vec{x}y),$$

(where the length of $\vec{x}$ is k) and consequently

$$HAP \vdash \tau_f\overline{a}_0\cdots\overline{a}_{n-1} = \overline{val(f,\vec{a})}$$

for every $a_0,\dots,a_{n-1} \in \mathbb{N}$. In particular $HAP \vdash \tau_f\downarrow$ for every such f.

Proof The proof is standard. See e.g. [10, Theorem 1.2.4] and [11, Theorem 9.3.11]. □

Fixing $\mathcal{L}_{HA}$ we can extend this interpretation compositionally to an interpretation $\mathcal{T}$ of $\mathcal{L}_{PRA}$ into $\mathcal{L}_{HAP}$, yielding the following result.

Proposition 3.10 *Recall $HA_{\mathcal{L}_{PRA}}$ is Heyting arithmetic axiomatised in $\mathcal{L}_{PRA}$. For every formula ϕ of $\mathcal{L}_{PRA}$,*

$$HA_{\mathcal{L}_{PRA}} \vdash \phi \Rightarrow HAP \vdash \mathcal{T}(\phi)$$

In particular, if $\phi \in \mathcal{L}_{PRA}$ is a true (in $\mathbb{N}$) Σ_1-sentence, then $HAP \vdash \mathcal{T}(\phi)$.

Proof Apply Lemma 3.6 to $\mathcal{T}[HA_{\mathcal{L}_{PRA}}] \subset HAP$. □

Proposition 3.11 (Recursion in HAP) *Combinatory algebra in* HAP *encompasses all recursive functions: there are combinator terms* **d***,* **fix** *and* **min** *such that*

$$\text{HAP} \vdash x = y \to \mathbf{d}xyfgz \simeq fz,$$
$$\text{HAP} \vdash x \neq y \to \mathbf{d}xyfgz \simeq gz,$$
$$\text{HAP} \vdash \mathbf{fix}\, f \downarrow \wedge\ \mathbf{fix}\, fx \simeq f(\mathbf{fix}\, f)x,$$
$$\text{HAP} \vdash \mathbf{min}\, fx \downarrow \leftrightarrow \exists y (fxy = 0 \wedge \forall z < y (fxz \downarrow))$$

and

$$\text{HAP} \vdash \mathbf{min}\, fx \downarrow \to (fx(\mathbf{min}\, fx) = 0 \wedge \forall y < (\mathbf{min}\, fx)(fxy > 0)).$$

Proof Again standard. See [10, Proposition 1.2.1.], [6, VI.2.6–8] and [11, 9.3.7–11]. □

The above thus embeds $\mathcal{L}_{\text{PRA}}$ (and more) into $\mathcal{L}_{\text{HAP}}$. It will, however, be important that $\mathcal{L}_{\text{HA}} \subset \mathcal{L}_{\text{PRA}}$ is embedded in $\mathcal{L}_{\text{HAP}}$ in a way that does not involve $\cdot$ and the pure combinators; recall that $\mathcal{L}_{\text{HA}} \subset \mathcal{L}_{\text{HAP}}$ *by definition* as the arithmetical formulae of $\mathcal{L}_{\text{HAP}}$, which are not affected by $\mathcal{T}$.

By interpreting $\cdot$ as Kleene application, we also have a result in the converse direction.

Proposition 3.12 *There is an interpretation* $\mathcal{F}$ *of* $\mathcal{L}_{\text{HAP}}$ *in* $\mathcal{L}_{\text{HA}}$ *which interprets* $\cdot$ *as Kleene application, such that*

1. $\text{HA} \vdash \phi \leftrightarrow \mathcal{F}(\phi)$ *for all* $\phi \in \mathcal{L}_{\text{HA}}$,
2. $\text{HAP} \vdash \phi \Rightarrow \text{HA} \vdash \mathcal{F}(\phi)$ *for all* $\phi \in \mathcal{L}_{\text{HAP}}$.

Corollary 3.13 (Proposition 2.4 of [7]) HAP *is conservative over* HA.

3.2 Realizability in HAP

Realizability in HAP is one of the main tools of [7], and we will utilise many of their results, and those of similar systems like in [6, 11]. We present this technical background, mainly without proof, in this subsection, together with some additional machinery.

We will, in fact, extend these definitions of realizability to languages $\mathcal{L}$ extending $\mathcal{L}_{\text{HAP}}$ with an unspecified set of relation symbols and parameterise this extension by the clauses for these symbols. We let $\mathcal{L}$ be fixed throughout this subsection.

Definition 3.14 *(Realizability)* For each term τ and formula φ we define the formula $\tau \underline{\mathbf{r}}\, \varphi$ after renaming of bound variables in φ to avoid conflicts with τ and nested bindings. The notion of realizability we will use is defined via the following clauses:

$$\tau \underline{r} (\sigma_1 = \sigma_2) \text{ is } \sigma_1 = \sigma_2$$
$$\tau \underline{r} \perp \text{ is } \perp$$
$$\tau \underline{r} (\phi \wedge \psi) \text{ is } (\mathbf{p}_\mathrm{l}\tau \underline{r} \phi) \wedge (\mathbf{p}_\mathrm{r}\tau \underline{r} \psi)$$
$$\tau \underline{r} (\phi \vee \psi) \text{ is } (\mathbf{p}_\mathrm{l}\tau = 0 \rightarrow \mathbf{p}_\mathrm{r}\tau \underline{r} \phi) \wedge (\mathbf{p}_\mathrm{l}\tau \neq 0 \rightarrow \mathbf{p}_\mathrm{r}\tau \underline{r} \psi)$$
$$\tau \underline{r} (\phi \rightarrow \psi) \text{ is } \forall y(y \underline{r} \phi \rightarrow (\tau y\downarrow \wedge \tau y \underline{r} \psi))$$
$$\tau \underline{r} \forall x\phi \text{ is } \forall x(\tau x\downarrow \wedge \tau x \underline{r} \phi)$$
$$\tau \underline{r} \exists x\phi \text{ is } \mathbf{p}_\mathrm{r}\tau \underline{r} (\phi(x/\mathbf{p}_\mathrm{l}\tau))$$
$$\tau \underline{r} R(\sigma_0, \dots, \sigma_{n-1}) \text{ is } Q_R(\sigma_0, \dots, \sigma_{n-1}, \tau)$$

where y is a fresh variable (which can be chosen in a systematic fashion which we will not specify). Here Q_R is an as yet unspecified relation symbol whose arity is one greater than that of R.

That this is a well-defined notion, in that it respects α-equivalence, can be proved by structural induction. Other standard properties of realizability can be proved in the same manner, e.g.

Lemma 3.15 *For every formula ϕ and term τ, $\tau \underline{r} \phi$ is negative and the free variables of ϕ are among the free variables of $\tau \underline{r} \phi$, which are among the free variables of ϕ and τ.*

For the rest of this section we will simply let $\underline{r}R$ denote the unspecified relation Q_R in the definition of $\underline{r}$ above, whence the final clause reads

$$\tau \underline{r} R(\sigma_0, \dots, \sigma_{n-1}) \text{ is } (\underline{r}R)(\sigma_0, \dots, \sigma_{n-1}, \tau)$$

for relations R. We will make some concrete choices for this $\underline{r}R$ in particular cases in Sect. 5.2. This draws attention to a potential subtlety, in that $\tau \underline{r}\varphi(\vec{x})$ could a priori mean both $\tau \underline{r} (\varphi(\vec{x}))$ and $(\tau \underline{r} \varphi)(\vec{x})$. As our choice of notation for the base case indicates, however, these are the same as long as τ contains none of the variables substituted for.

Lemma 3.16 (Lemma 4.4.6.(ii) of [8]) *Substitution distributes over realizability: Let φ be a formula and τ a term. For all equally long finite sequences $\vec{u}$ and $\vec{\sigma}$ of distinct variables and terms, respectively, $(\tau \underline{r}\varphi)(\vec{u}/\vec{\sigma}) = (\tau[\vec{u}/\vec{\sigma}]) \underline{r} (\varphi(\vec{u}/\vec{\sigma}))$. In particular, $(x \underline{r} \varphi)(x/\tau) = \tau \underline{r} \varphi$ if x is fresh.*

To disambiguate when there is a potential clash we will let $\tau \underline{r}\varphi(\vec{y})$ mean $\tau \underline{r} (\varphi(\vec{y}))$ etc., since this seems to be the natural reading when τ is a(n explicitly given) single variable.

The relationship between realizability and substitution of formulae for relation symbols is only slightly more involved, since the former might change the relations occurring in a formula.

Lemma 3.17 *Let $n \in \mathbb{N}$, R be an n-ary relation symbol, τ be a term and ϑ be a formula without free occurrences of v_n. Suppose $\underline{\mathsf{r}}R$ is not the same as $\underline{\mathsf{r}}Q$ for any other relation symbol Q. Then $\tau \underline{\mathsf{r}} (\varphi(R/\lambda\vec{\mathsf{v}}_{<n}.\vartheta)) = (\tau \underline{\mathsf{r}} \varphi)(\underline{\mathsf{r}}R/\lambda\vec{\mathsf{v}}_{\leq n}.\mathsf{v}_n \underline{\mathsf{r}} \vartheta)$ for every $\varphi \in \mathcal{L}$.*

Proof Induction on the construction of φ.

- All base-cases except $R(\vec{\sigma})$ follow since the substitutions are vacuous. Moreover

$$
\begin{aligned}
\tau \underline{\mathsf{r}} (R(\vec{\sigma})(R/\lambda\vec{\mathsf{v}}_{<n}.\vartheta)) &= \tau \underline{\mathsf{r}} (\vartheta(\vec{\mathsf{v}}_{<n}/\vec{\sigma})) \\
&= (\mathsf{v}_n \underline{\mathsf{r}} \vartheta)(\vec{\mathsf{v}}_{\leq n}/\vec{\sigma}, \tau) \\
&= (\underline{\mathsf{r}}R(\vec{\sigma}, \tau))(\underline{\mathsf{r}}R/\lambda\vec{\mathsf{v}}_{\leq n}.\mathsf{v}_n \underline{\mathsf{r}} \vartheta) \\
&= (\tau \underline{\mathsf{r}} R(\vec{\sigma}))(\underline{\mathsf{r}}R/\lambda\vec{\mathsf{v}}_{\leq n}.\mathsf{v}_n \underline{\mathsf{r}} \vartheta)
\end{aligned}
$$

  for all terms τ by Lemma 3.16.
- Assume the statement holds for χ and ψ and let τ be a term. Then

$$
\begin{aligned}
\tau \underline{\mathsf{r}} &((\chi \wedge \psi)(R/\lambda\vec{\mathsf{v}}_{<n}.\vartheta)) \\
&= \mathbf{p}_\mathsf{l}\tau \underline{\mathsf{r}} (\chi(R/\lambda\vec{\mathsf{v}}_{<n}.\vartheta)) \wedge \mathbf{p}_\mathsf{r}\tau \underline{\mathsf{r}} (\psi(R/\lambda\vec{\mathsf{v}}_{<n}.\vartheta)) \\
&= (\mathbf{p}_\mathsf{l}\tau \underline{\mathsf{r}} \chi)(\underline{\mathsf{r}}R/\lambda\vec{\mathsf{v}}_{\leq n}.\mathsf{v}_n \underline{\mathsf{r}} \vartheta) \wedge (\mathbf{p}_\mathsf{r}\tau \underline{\mathsf{r}} \psi)(\underline{\mathsf{r}}R/\lambda\vec{\mathsf{v}}_{\leq n}.\mathsf{v}_n \underline{\mathsf{r}} \vartheta) \\
&= (\tau \underline{\mathsf{r}} (\chi \wedge \psi))(\underline{\mathsf{r}}R/\lambda\vec{\mathsf{v}}_{\leq n}.\mathsf{v}_n \underline{\mathsf{r}} \vartheta)
\end{aligned}
$$

  by induction hypothesis. The disjunctive case is similar.
- Assume the statement holds for χ and ψ and let τ be a term and y a variable without free occurrences in χ, ψ, ϑ or τ. Then

$$
\begin{aligned}
\tau \underline{\mathsf{r}} &((\chi \rightarrow \psi)(R/\lambda\vec{\mathsf{v}}_{<n}.\vartheta)) \\
&= \forall y(y \underline{\mathsf{r}} (\chi(R/\lambda\vec{\mathsf{v}}_{<n}.\vartheta)) \rightarrow (\tau y{\downarrow} \wedge \tau y \underline{\mathsf{r}} (\psi(R/\lambda\vec{\mathsf{v}}_{<n}.\vartheta)))) \\
&= \forall y((y \underline{\mathsf{r}} \chi)(\underline{\mathsf{r}}R/\lambda\vec{\mathsf{v}}_{\leq n}.\mathsf{v}_n \underline{\mathsf{r}} \vartheta) \rightarrow (\tau y{\downarrow} \wedge (\tau y \underline{\mathsf{r}} \psi)(\underline{\mathsf{r}}R/\lambda\vec{\mathsf{v}}_{\leq n}.\mathsf{v}_n \underline{\mathsf{r}} \vartheta))) \\
&= (\tau \underline{\mathsf{r}} (\chi \rightarrow \psi))(\underline{\mathsf{r}}R/\lambda\vec{\mathsf{v}}_{\leq n}.\mathsf{v}_n \underline{\mathsf{r}} \vartheta)
\end{aligned}
$$

  by induction hypothesis and since y is fresh.
- Assume the statement holds for ψ and let τ be a term. We can without loss of generality assume that x does not occur in τ or ϑ. Then

$$
\begin{aligned}
\tau \underline{\mathsf{r}} ((\forall x\psi)(R/\lambda\vec{\mathsf{v}}_{<n}.\vartheta)) &= \forall x(\tau x{\downarrow} \wedge \tau x \underline{\mathsf{r}} (\psi(R/\lambda\vec{\mathsf{v}}_{<n}.\vartheta))) \\
&= \forall x(\tau x{\downarrow} \wedge ((\tau x \underline{\mathsf{r}} \psi)(\underline{\mathsf{r}}R/\lambda\vec{\mathsf{v}}_{\leq n}.\mathsf{v}_n \underline{\mathsf{r}} \vartheta))) \\
&= (\tau \underline{\mathsf{r}} \forall x\psi)(\underline{\mathsf{r}}R/\lambda\vec{\mathsf{v}}_{\leq n}.\mathsf{v}_n \underline{\mathsf{r}} \vartheta)
\end{aligned}
$$

  by induction hypothesis.

- Assume the statement holds of ψ and let τ be a term. We can without loss of generality assume that x does not occur in τ or ϑ. Then

$$
\begin{aligned}
\tau \underline{r} ((\exists x \psi)(R/\lambda \vec{v}_{<n}.\vartheta)) &= \mathbf{p}_r \tau \underline{r} (\psi (R/\lambda \vec{v}_{<n}.\vartheta)(x/\mathbf{p}_l \tau)) \\
&= \mathbf{p}_r \tau \underline{r} (\psi (x/\mathbf{p}_l \tau)(R/\lambda \vec{v}_{<n}.\vartheta)) \\
&= (\mathbf{p}_r \tau \underline{r} (\psi (x/\mathbf{p}_l \tau)))(\underline{r} R/\lambda \vec{v}_{\leq n}.v_n \underline{r} \vartheta) \\
&= (\tau \underline{r} (\exists x \psi))(\underline{r} R/\lambda \vec{v}_{\leq n}.v_n \underline{r} \vartheta)
\end{aligned}
$$

by induction hypothesis.

This concludes the induction. □

Lemma 3.18 *Realizability is closed under derivability in* LPT: *Let Δ be a set of formulae including the axioms of* HAP($\mathcal{L}$) *and Γ be a set of formulae such that for each $\gamma \in \Gamma$ there is a term σ with free variables among those of γ and $\Delta \vdash \sigma \downarrow \wedge \sigma \underline{r} \gamma$. Let φ be a formula and $\mathscr{D}$ a derivation with $\Gamma \vdash^{\varnothing} \varphi$. Then there is a term τ with free variables among those of φ such that $\Delta \vdash \tau \downarrow \wedge \tau \underline{r} \varphi$.*

Proof By induction on the depth of $\mathscr{D}$. The proof is similar to [6, VII. Theorem 1.6].
□

While we rely heavily on the work in [7], our definitions of realizability are not literally the same, since there disjunction is a defined predicate ($\varphi \vee \psi$ is $\exists n((n = 0 \rightarrow \varphi) \wedge (n \neq 0 \rightarrow \psi)))$. This difference is immaterial, however, since the notions are equivalent in the following sense. Let $\underline{\hat{r}}$ denote the notion of realizability from [7].

Lemma 3.19 *The realizability notion $\underline{r}$ is equivalent to $\underline{\hat{r}}$: for every formula $\varphi \in \mathcal{L}_{HAP}$ there are unary total combinators μ and ν such that* HAP $\vdash \forall x(x \underline{r} \varphi \rightarrow \mu \cdot x \underline{\hat{r}} \varphi)$ *and* HAP $\vdash \forall x(x \underline{\hat{r}} \varphi \rightarrow \nu \cdot x \underline{r} \varphi)$. *Consequently* HAP $\vdash \exists x(x \underline{r} \varphi) \leftrightarrow \exists x(x \underline{\hat{r}} \varphi)$.

Since we can thus identify the two notions of realizability we get the following as a consequence of Corollary 3.6 from [7]:

Proposition 3.20 HAP *plus the schema $\varphi \leftrightarrow \exists x(x \underline{r} \varphi)$ for $\varphi \in \mathcal{L}_{HA}$ is conservative over* HA. *In particular, for every formula $\varphi \in \mathcal{L}_{HA}$, if* HAP $\vdash \exists x(x \underline{r} \varphi)$ *then* HA $\vdash \varphi$.

It is important for the above proposition that $\mathcal{L}_{HA}$ is a subset of $\mathcal{L}_{HAP}$, instead of being embedded into it in a way involving the (pure) combinators or the application function symbol specific to the language $\mathcal{L}_{HAP}$ (like the inclusion of the rest of $\mathcal{L}_{PRA}$ in Proposition 3.9). In addition to allowing us to state the theorem as a simple conservativity result, this is required for the proof of [7, Corollary 3.6], which relies on an interpretation of HAP into itself which changes the application function and pure combinators, while preserving $\mathcal{L}_{HA}$.

4 A hierarchy of almost negative formulae

An important subtheory of $\widehat{\mathrm{ID}}_1^i$ is that given by the fixpoint axioms for operator forms which are almost negative. Recall a formula is almost negative if it contains no disjunctions and the matrix of any existential quantifier is an equation between terms. Λ denotes the class of almost negative formulae.

In this section we introduce a hierarchy (Λ_n) of formulae which exhausts the almost negative formulae Λ. In classical arithmetic, the arithmetic (quantifier) hierarchy classifies formulae of the language according by quantifier complexity (that is, quantifier alternations). This hinges on the existence of a prenex normal form, and in particular on De Morgan's laws and the possibility to define $\rightarrow$ in terms of $\vee$ and $\neg$. For intuitionistic arithmetic this is no longer a possibility and the situation is not as simple. A few hierarchies for intuitionistic arithmetic have been proposed, in this context we should mention Wolfgang Burr whose Φ'-hierarchy in [12] is very close to the one we consider (see also Burr's [13]).

This formula hierarchy will behave relative to Λ in several respects similarly to how the arithmetic hierarchy behaves relative to the entire language in the classical case. In addition to exhausting the almost negative formulae (in a sense made precise below), each level of the hierarchy contains a partial satisfaction predicate for that level: for each $n \in \mathbb{N}$ there is a Λ_n-formula $\mathtt{Sat}_n$ which HAP proves to be a (compositional) satisfaction predicate for formulae in Λ_n. The construction of the $\mathtt{Sat}_n$ will be the main content of Sect. 4.2.

4.1 Stratifying the almost negative formulae

For now we turn to more elementary properties of Λ_n. Two of the most notable of these are the Diagonal Lemma 4.6 and the closure of Λ_n under substitution into strictly positive positions (Lemma 4.5). These properties play crucial roles in the interpretability argument of Theorem 5.7.

Definition 4.1 Σe, the set of Σ-equations, is the set of formulae of the form $\exists x \, \sigma = \tau$, where σ and τ are terms. Λ_0 is the smallest set closed under conjunction which contains Σe and the atomic formulae. Λ_{n+1} is the smallest set that contains Σe and atomic formulae which is closed under conjunction, universal quantification and implications with Λ_n antecedents; that is, if $\varphi \in \Lambda_{n+1}$ and $\psi \in \Lambda_n$ then $\psi \rightarrow \varphi \in \Lambda_{n+1}$.

The normal form hierarchy $(\Lambda_n^{\mathrm{nf}})_n$ is defined as follows: Λ_0^{nf} is the set conjunctions $\psi \wedge \bigwedge_{i=1}^k \varphi_i$ where $\psi \in \Sigma$e, $k \geq 0$ and each φ_i for $i \leq k$ is an atomic formula other than $\bot$ or an equation (whence $\Lambda_0^{\mathrm{nf}} = \Sigma$e for $\mathcal{L}_{\mathrm{HA}}$, $\mathcal{L}_{\mathrm{PRA}}$ and $\mathcal{L}_{\mathrm{HAP}}$). $\Lambda_{n+1}^{\mathrm{nf}}$ is the set of universally quantified conjunctions of implications with Λ_n^{nf} antecedents and Λ_0^{nf} consequents, namely formulae of the form $\forall x \bigwedge_{i=0}^k (\phi_i \rightarrow \psi_i)$ where $\{\phi_0, \ldots, \phi_k\} \subseteq \Lambda_n^{\mathrm{nf}}$ and $\{\psi_0, \ldots, \psi_k\} \subseteq \Lambda_0^{\mathrm{nf}}$. Let $\Lambda^{\mathrm{nf}} = \bigcup_{n \in \mathbb{N}} \Lambda_n^{\mathrm{nf}}$.

It should be noted that this stratification of Λ is closed under α-equivalence and thus is well defined. A consequence of the definition is that the Λ_n form a strictly telescoping hierarchy:

Lemma 4.2 *For every $n \in \mathbb{N}$, $\Lambda_n \subset \Lambda_{n+1}$.*

The hierarchy $(\Lambda_n^{\mathrm{nf}})_n$ represents normal forms for formulae of Λ in the sense that $\Lambda_n^{\mathrm{nf}} \subseteq \Lambda_n$ for every n, and every formula in Λ_n is provably equivalent to a formula in Λ_n^{nf} over the theory HA or HAP as appropriate. This latter result is strengthened in Lemma 4.4.

We are interested here in the almost negative formulae, which is why we allow Σe as 'atoms,' much as Δ_0-formulae are treated as 'atoms' in the classical arithmetical hierarchy. In the latter case, however, we have generous closure conditions on this set of 'atoms,' even though this makes no real difference further on in the arithmetical hierarchy, while here we are requiring the special 'atoms' to have a very concrete form. We could, for the purposes of this paper, have let Σe be closed under (at least) conjunctions, disjunctions and existential quantification and subsequently used formulae constructed therefrom by conjunctions, implications and universal quantifications instead of the almost negative formulae (thus thereby essentially allowing general Σ_1-formulae as 'atoms') without much extra work. Since the almost negative formulae seem to be the established standard in these contexts, however, we will not generalise in this direction. This will have the added advantage of making the definitions of the satisfaction predicates somewhat clearer since no connectives or quantifiers are in the scope of an existential quantifier.

In the other direction, as opposed to Heyting arithmetic in the language of PRA, every (partial) recursive function is represented by a term in HAP. Thus, adding an existential quantifier to form a Σ-equation does not yield increased expressibility. So we could have simplified the definition of Λ^{nf} in this setting by leaving out the existential quantifiers altogether. However, we gain little by doing so, and thus prefer to have Λ^{nf} formulated as a template for all adequate languages.

The essential difference between the Λ-hierarchy and Burr's Φ' from [12] is that Λ_n is also closed under conjunction to exhaust the almost negative formulae. Both can be seen as mirroring the ordinary arithmetic hierarchy, in that they are classically equivalent to it.

In the following lemma, let PAP denote the result of adding the law of excluded middle to HAP.[3] Given a set Γ of formulae and a theory T, recall $\Gamma(\mathrm{T})$ denotes the set of formulae that are provably in T equivalent to some formula in Γ.

Lemma 4.3 Λ_n *relates to almost negative formulae and the usual arithmetical hierarchy in the following way:*

1. *The Λ_n exhaust the almost negative formulae, that is $\Lambda = \bigcup_{n \in \mathbb{N}} \Lambda_n$.*
2. *In the arithmetic languages $\mathcal{L}_{\mathrm{PRA}}$ and $\mathcal{L}_{\mathrm{HAP}}$: Let HA^* be $\mathrm{HA}_{\mathcal{L}_{\mathrm{PRA}}}$ or HAP. Then $\Lambda_0^{\mathrm{nf}}(\mathrm{HA}^*)$, the formulae equivalent to Λ_0^{nf}-formulae over HA^*, is precisely the formulae equivalent to Σe-formulae over HA^*, as well as the formulae equivalent to Σ_1-formulae over HA^*.*
3. *In the arithmetic languages $\mathcal{L}_{\mathrm{PRA}}$ and $\mathcal{L}_{\mathrm{HAP}}$: Let PA^* be $\mathrm{PA}_{\mathcal{L}_{\mathrm{PRA}}}$ or PAP. For $n > 0$, $\Lambda_n^{\mathrm{nf}}(\mathrm{PA}^*) = \Pi_{n+1}(\mathrm{PA}^*)$. Moreover, $\Lambda_0^{\mathrm{nf}}(\mathrm{PA}^*) = \Sigma\mathrm{e}(\mathrm{PA}^*) = \Sigma_1(\mathrm{PA}^*)$.*

Proof Observe that $\Lambda_n \subseteq \Lambda$ for all $n \in \mathbb{N}$. We show that $\Lambda \subseteq \bigcup_n \Lambda_n$ by induction on the complexity of almost negative formulae:

- If φ is an atom or a Σe-formula, then $\varphi \in \Lambda_0$.
- The induction steps for $\wedge$ and $\forall$ are straightforward.
- If $\varphi = (\vartheta \to \psi)$ for $\vartheta, \psi \in \bigcup_{k \in \mathbb{N}} \Lambda_k \cap \Lambda$, let n be such that both are in Λ_n (by Lemma 4.2). Then $\varphi \in \Lambda_{n+1}$.

---

[3] To the best of our knowledge the theory PAP has not been studied. We thus make no claims on suitability for any particular purpose, but merely use it as a name for adding LEM to HAP.

This concludes the proof of the first statement.

For the remaining statements, clearly $\Sigma e = \Lambda_0^{nf} \subseteq \Sigma_1$ by definition. We show by induction on $n > 0$ that $\Lambda_n^{nf} \subseteq \Pi_{n+1}(PA^*)$.

- Let $\varphi = \forall x \bigwedge_i (\vartheta_i \to \psi_i)$, where $\vartheta_i, \psi_i \in \Lambda_0^{nf} \subseteq \Sigma_1$. Then by classical logic $\varphi \in \Pi_2(PA^*)$.
- Let $k \geq 1$ and $\varphi = \forall x \bigwedge_i (\vartheta_i \to \psi_i)$, where $\vartheta_i \in \Lambda_k^{nf} \subseteq \Pi_{k+1}(PA^*)$ and $\psi_i \in \Lambda_0^{nf} \subseteq \Sigma_1$. Then by classical logic $\varphi \in \Pi_{k+2}(PA^*)$.

Conversely, since Δ_0-formulae in $\mathcal{L}_{PRA}$ and $\mathcal{L}_{HAP}$ are equivalent to atoms in HA*, $\Sigma_1 \subseteq \Sigma e(HA^*) = \Lambda_0^{nf}(HA^*) \subseteq \Lambda_0^{nf}(PA^*)$ by structural induction. Hence $\Sigma_1(HA^*) = \Lambda_0^{nf}(HA^*)$ and the corresponding for PA*. We show that $\Pi_{n+1}(PA^*) = \Lambda_n^{nf}(PA^*)$ by induction on $n > 0$.

- Suppose $\varphi = \forall x \vartheta$, where $\vartheta \in \Sigma_1 \subseteq \Lambda_0^{nf}(PA^*)$. Then $\varphi \in \Lambda_1^{nf}(PA^*)$. Thus $\Pi_2 \subseteq \Lambda_1^{nf}(PA^*)$, whence $\Pi_2(PA^*) = \Lambda_1^{nf}(PA^*)$ by above.
- Suppose $\varphi = \forall x \vartheta$, where $\vartheta \in \Sigma_{k+1}$. Let $\dot{\neg} \chi$ be $\chi \to \exists x (0 = S(0))$ for formulae χ. Then $\dot{\neg} \vartheta \in \Pi_{k+1}(PA^*) = \Lambda_k^{nf}(PA^*)$ by induction hypothesis. Let $\psi \in \Lambda_k^{nf}$ be equivalent. Then $PA^* \vdash \varphi \leftrightarrow \forall x \dot{\neg} \psi$, where $\forall x \dot{\neg} \psi \in \Lambda_{k+1}^{nf}$. Note that this would not work with $\neg$, since $\perp$ is not a Λ_0^{nf}-formula. So $\Pi_{k+2} \subseteq \Lambda_{k+1}^{nf}(PA^*)$ and consequently $\Pi_{k+2}(PA^*) = \Lambda_{k+1}^{nf}(PA^*)$. □

In particular, $\Sigma e(HAP)$ is closed under conjunction, disjunction, existential quantification and bounded universal quantification.

It should be intuitively clear that Λ_0 is HAP-equivalent to Λ_0^{nf}. The same holds for all n, as we will show. In order to define a satisfaction predicate for Λ_n, however, we need to have access to an explicit transformation from Λ_n to its normal form Λ_n^{nf} in HAP. Corresponding transformations can be defined for languages other than $\mathcal{L}_{HAP}$. Since it is clear that $\Lambda_n^{nf} \subseteq \Lambda_n$, this transformation yields the inclusions $\Lambda_n^{nf} \subseteq \Lambda_n \subseteq \Lambda_n^{nf}(HAP)$ and $\Lambda_n^{nf} \subseteq \Lambda_n \subseteq \Lambda_n^{nf}(PAP)$ for all n.

We now fix a standard coding of the language $\mathcal{L}_{HAP}$ in HA, and thus in HAP. The features we will assume are that the set of codes of two types of syntactic objects (terms, formulae etc.) never intersect, that no two distinct syntactic objects are assigned the same code, that 0 is not the code of any syntactic object and that the code of formulae and terms are (numerically) strictly larger than their (immediate) constituents. A similar note on the coding of sequences are that they are assumed to be strictly larger than their length and all of their elements. Thus we can reason about Λ_n-formulae in a theory of arithmetic.

Lemma 4.4 *There are primitive recursive functions* $\lambda_n \colon \Lambda_n \longrightarrow \Lambda_n^{nf}$ *which transforms* Λ_n-*formulae to* HAP-*provably equivalent normal forms.*

Proof We omit the actual construction, from which we can see that $HAP \vdash \varphi \leftrightarrow \lambda_n(\varphi)$ for $\varphi \in \Lambda_n$ but which is otherwise unenlightening. We only give the general ideas on which it hinges.

Firstly, we can easily add redundant parts to a $(\Sigma\text{-})$equation to make it an equivalent Λ_n^{nf}-formula, as well as finding a Σ-equation equivalent to $\perp$ (as above).

Secondly, if $\exists x(\sigma_1 = \sigma_2)$ and $\exists y(\tau_1 = \tau_2)$ are Λ_0^{nf}-formulae, then $\exists v(((\sigma_1 \dotdiv \sigma_2) + (\sigma_2 \dotdiv \sigma_1))(x/\mathbf{p}_l v) + ((\tau_1 \dotdiv \tau_2) + (\tau_2 \dotdiv \tau_1))(y/\mathbf{p}_r v)) = 0)$ is a Λ_0^{nf}-formula equivalent to their conjunction (where $\dotdiv$ is a shorthand for application of the combinator representing the primitive recursive subtraction function).

Thirdly, if $\forall x \bigwedge_{i=0}^{k}(\psi_i \to \varphi_i)$ and $\forall y \bigwedge_{j=0}^{m}(\xi_j \to \vartheta_j)$ are $\Lambda_{n+1}^{\mathrm{nf}}$-formulae and $\chi \in \Lambda_n^{\mathrm{nf}}$, then

$$\forall x \bigwedge_{i=0}^{k}(\psi_i \to \varphi_i) \wedge \forall y \bigwedge_{j=0}^{m}(\xi_j \to \vartheta_j)$$

$$\forall u \forall x \bigwedge_{i=0}^{k}(\psi_i \to \varphi_i)$$

and

$$\chi \to \forall x \bigwedge_{i=0}^{k}(\psi_i \to \varphi_i)$$

where x, y and u could be the same, are equivalent to

$$\forall v((\bigwedge_{i=0}^{k}\psi_i \to \varphi_i)(x/v) \wedge (\bigwedge_{j=0}^{m}\xi_j \to \vartheta_j)(y/v))$$

$$\forall v(\bigwedge_{i=0}^{k}(\psi_i \to \varphi_i)(x/\mathbf{p}_l v)(u/\mathbf{p}_r v))$$

and

$$\forall v \bigwedge_{i=0}^{k}((\chi \wedge \psi_i(x/v)) \to \varphi_i(x/v))$$

respectively, where v is fresh. The first two are in $\Lambda_{n+1}^{\mathrm{nf}}$ and the third can be given such a form inductively, since $\chi \wedge \psi_i(x/v) \in \Lambda_n$. $\qquad\square$

The λ_n can be chosen to be (provably in HAP) primitive recursive in the structure of their arguments.

We finally turn to the two key properties of the hierarchy mentioned earlier.

Lemma 4.5 *If $\phi \in \Lambda_n$ has only strictly positive occurrences of the k-ary relation symbol R and $\theta \in \Lambda_m$ then $\phi(R/\theta) \in \Lambda_{\max(n,m)}$.*

Proof Let $M = \max(n, m)$. The proof is by induction on the complexity of ϕ.

- Let σ and τ be terms. Then $(\sigma = \tau)(R/\theta) = (\sigma = \tau) \in \Lambda_M$.
- We have $\bot(R/\theta) = \bot \in \Lambda_M$.

- We have $(R(\vec{\tau}))(R/\theta) = \theta(\vec{v}_{<k}/\vec{\tau}) \in \Lambda_M$.
- We have $(Q(\vec{\tau}))(R/\theta) = Q(\vec{\tau}) \in \Lambda_M$.
- Let σ and τ be terms. Then $(\exists x \sigma = \tau)(R/\theta) = (\exists x \sigma = \tau) \in \Lambda_M$.
- Let $\chi, \psi \in \Lambda_n$ have only strictly positive occurrences of R and be such that $\chi(R/\theta), \psi(R/\theta) \in \Lambda_M$. Then $(\chi \wedge \psi)(R/\theta) = (\chi(R/\theta) \wedge \psi(R/\theta)) \in \Lambda_M$.
- In case $n \geq 1$, let $\psi \in \Lambda_n$ have only strictly positive occurrences of R and be such that $\psi(x/z)(R/\theta) \in \Lambda_M$ for all z. Then $(\forall x \psi)(R/\theta) = \forall y(\psi(x/y)(R/\theta)) \in \Lambda_M$ for some fresh y.
- In case $n \geq 1$, let $\psi \in \Lambda_n$ have only strictly positive occurrences of R and be such that $\psi(R/\theta) \in \Lambda_M$, and $\chi \in \Lambda_{n-1}$ be such that $\chi \to \psi$ has only strictly positive occurrences of R. Then $(\chi \to \psi)(R/\theta) = (\chi \to \psi(R/\theta)) \in \Lambda_M$ since R does not occur in χ.

This concludes the proof. $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad$ □

Since HAP has terms for all primitive recursive functions, there is a term `sub` for the substitution function in $\mathcal{L}_{\mathrm{HAP}}$. This is used to prove the following:

Lemma 4.6 (Diagonal Lemma in HAP) Λ_n and Λ_n^{nf} are closed under diagonalisation: If $\varphi \in \Lambda_n$ (resp. $\in \Lambda_n^{\mathrm{nf}}$) has free variables $v_0, \ldots, v_k$, then there is a formula $\psi \in \Lambda_n$ ($\psi \in \Lambda_n^{\mathrm{nf}}$) with free variables $v_0, \ldots, v_{k-1}$ and

$$\mathrm{HAP} \vdash \psi(\vec{x}) \leftrightarrow \varphi(\vec{x}, \ulcorner \psi \urcorner),$$

where $\vec{x}$ has length k.

Proof This is the standard diagonal argument. $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad$ □

4.2 Satisfaction predicates for almost negative formulae

Given terms η, v and τ we write η_v^τ for the term $\lambda u.\mathbf{d}uv(\mathbf{k}\tau)\eta u$, where u is fresh. (Recall $\mathbf{d}$ is the 'decision by case' term from Proposition 3.11.) This term is such that $\mathrm{HAP} \vdash \forall xy(\eta_x^\tau \cdot x \simeq \tau \wedge (x \neq y \to \eta_x^\tau \cdot y \simeq \eta \cdot y))$, corresponding to the 'update' of the function η with the new value τ for input v. Note, there is no assumption that τ denotes. For sequences $\vec{\tau}$ and $\vec{v}$ of terms (of equal length) we define $\eta_{\vec{v}}^{\vec{\tau}}$ by iterating the above construction.

Since partial combinatory algebras are Turing complete (see Proposition 3.11 and [6, VI. Theorem 2.8.1]) it is possible to define a valuation combinator for partial terms of $\mathcal{L}_{\mathrm{HAP}}$.

Definition 4.7 (Valuation function in HAP) Using Proposition 3.9 we have formulae and combinators corresponding to the following primitive recursive relations and functions: `Seq`, `lh`, `Var`, `FV`, `BV` (the second argument is a (code of a) free/bound variable of the first), `Term`, `Fmla`, `Sent`, `sub`, Σe, Λ_n^{nf}, λ_n and `num` (the formal numeral function).

Let `val` be a combinator in HAP such that

1. $\mathrm{HAP} \vdash \mathtt{val} \cdot e \cdot \ulcorner \mathsf{c} \urcorner = \mathsf{c}$ for every constant symbol c,
2. $\mathrm{HAP} \vdash \mathtt{Var}(x) \rightarrow \mathtt{val} \cdot e \cdot x \simeq e \cdot x$, and
3. $\mathrm{HAP} \vdash \mathtt{val} \cdot e \cdot (\ulcorner \mathsf{f} \urcorner (\vec{a})) \simeq \mathsf{f}(\overrightarrow{\mathtt{val} \cdot e \cdot a})$ for every function symbol f

(recall the notational convention that $\ulcorner \alpha \urcorner$ means $\overline{\ulcorner \alpha \urcorner}$, the numeral of the Gödel code of α, for expressions α).

These defined relations will notationally be treated as predicate symbols.

Recall that in HAP every object in the domain is a partial function and, in particular, is a partial variable assignment mapping codes of variables to values. Thus it is not necessary to place any restriction on the e in $\mathtt{val} \cdot e \cdot z$ such as, for example, that e represents an assignment for the variables of (the term coded by the value of the variable) z.

Lemma 4.8 *The valuation function* $\mathtt{val}$ *acts as a partial valuation function for partial terms of* $\mathcal{L}_{\mathrm{HAP}}$. *In particular:*

1. $\mathrm{HAP} \vdash \forall e, f, t((\mathtt{Term}(t) \wedge \forall x < t(\mathtt{FV}(t, x) \rightarrow e \cdot x \simeq f \cdot x)) \rightarrow \mathtt{val} \cdot e \cdot t \simeq \mathtt{val} \cdot f \cdot t)$.
2. $\mathrm{HAP} \vdash \forall e, t, v, x(\mathtt{Term}(t) \wedge \neg \mathtt{FV}(t, v) \rightarrow \mathtt{val} \cdot e_v^x \cdot t \simeq \mathtt{val} \cdot e \cdot t)$.
3. $\mathrm{HAP} \vdash \forall e, s, t, x(\mathtt{Var}(x) \wedge \mathtt{Term}(t) \wedge \mathtt{Term}(s) \rightarrow \mathtt{val} \cdot e \cdot (\mathtt{sub} \cdot t \cdot x \cdot s) \simeq \mathtt{val} \cdot e_x^{\mathtt{val} \cdot e \cdot s} \cdot t)$.
4. $\mathrm{HAP} \vdash \forall e(\mathtt{val} \cdot e \cdot \ulcorner \tau \urcorner \simeq \tau(e \cdot \ulcorner \mathsf{v}_0 \urcorner, \ldots, e \cdot \ulcorner \mathsf{v}_{n-1} \urcorner))$ *for every partial term* τ *with free variables among* $\mathsf{v}_0, \ldots, \mathsf{v}_{n-1}$.

Proof The first three are proved by induction on t in HAP. The final claim is an induction on the construction of τ. □

In the interest of readability we will conflate coding and concatenation in the coded language. Any logical symbols occurring within square brackets [ ] refer to the codes of these symbols, concatenated with the other (named) constructs appearing in the brackets. For instance, $[\exists v(s = t)]$ below is the code $\ulcorner \exists \mathsf{v}_0(\mathsf{v}_1 = \mathsf{v}_2) \urcorner$ with the codes v, s, t formally substituted for $\ulcorner \mathsf{v}_0 \urcorner, \ulcorner \mathsf{v}_1 \urcorner, \ulcorner \mathsf{v}_2 \urcorner$ respectively.

Theorem 4.9 (Partial Satisfaction Predicates in HAP) *For every n there is a Λ_n^{nf}-formula* $\mathtt{Sat}_n(e, F)$ *which* HAP *proves is a compositional satisfaction predicate for Λ_n-formulae.*

Proof The proof is cumbersome but straightforward. Essentially, we mimic standard constructions of Σ_n-satisfaction predicates in PA, as in e.g. Hájek and Pudlák's [14].

We first construct an intermediate satisfaction predicate for Λ_n^{nf}-formulae in $\mathcal{L}_{\mathrm{HAP}}$ by induction on n.

- First $\mathtt{Sat}_0''(e, F)$ is

$$\exists x, v, s, t(\mathtt{Term}(s) \wedge \mathtt{Term}(t) \wedge \mathtt{Var}(v) \wedge F = [\exists v(s = t)] \wedge \mathtt{val} \cdot e_v^x \cdot s = \mathtt{val} \cdot e_v^x \cdot t)$$

This is straightforwardly HAP-equivalent to a Σe-formula. The predicate $\mathtt{Sat}_0'$ is thus the Λ_0^{nf} normal form of this formula, which is HAP-equivalent to it by Lemma 4.4.

- We define $\text{Sat}''_{n+1}(e, F)$ as

$$\Lambda^{\text{nf}}_{n+1}(F) \wedge \forall x, v, G, s(\text{Var}(v) \wedge F = [\forall v G]$$
$$\wedge \text{Seq}(s) \wedge \text{lh} \cdot s > 0 \wedge G = \text{conjs} \cdot s$$
$$\rightarrow \forall i < \text{lh} \cdot s \, \forall f, g < s_i(s_i = [g \rightarrow f] \wedge \Lambda^{\text{nf}}_n(g) \wedge \Lambda^{\text{nf}}_0(f)$$
$$\rightarrow (\text{Sat}'_n(e^x_v, g) \rightarrow \text{Sat}'_0(e^x_v, f))))$$

where conjs is the combinator representing the primitive recursive function

$$\text{conjs}(\langle e \rangle) = e$$
$$\text{conjs}(\langle h, t \rangle) = \text{conjs}(t) \wedge h,$$

in accordance with Proposition 3.9.

Sat''_{n+1} is Λ_{n+1}, whence Sat'_{n+1} is the $\Lambda^{\text{nf}}_{n+1}$-formula $\lambda_{n+1}(\text{Sat}''_{n+1})$.

We now define $\text{Sat}_n(e, F)$ as $\text{Sat}'_n(e, \lambda_n \cdot F)$, which is compositional on Λ_n-formulae. This, and the intermediate facts that Sat'_n respects conjunction, only depends on the values of e for the free variables of F and 'commutes' with substitution, that is

$$\text{HAP} \vdash \forall F, G, e(\Lambda^{\text{nf}}_n(F) \wedge \Lambda^{\text{nf}}_n(G)$$
$$\rightarrow (\text{Sat}'_n(e, \lambda_n \cdot [F \wedge G]) \leftrightarrow \text{Sat}'_n(e, F) \wedge \text{Sat}'_n(e, G))),$$
$$\text{HAP} \vdash \forall F, e, f(\Lambda^{\text{nf}}_n(F) \wedge \forall x < F(\text{FV}(F, x) \rightarrow e \cdot x \simeq f \cdot x)$$
$$\rightarrow (\text{Sat}'_n(e, F) \leftrightarrow \text{Sat}'_n(f, F)))$$

and

$$\text{HAP} \vdash \forall F, x, s, e(\Lambda^{\text{nf}}_n(F) \wedge \text{Var}(x) \wedge \text{Term}(s) \wedge (\forall v < F \neg(\text{BV}(F, v) \wedge \text{FV}(s, v)))$$
$$\rightarrow (\text{Sat}'_n(e, \text{sub} \cdot F \cdot x \cdot s) \leftrightarrow \text{Sat}'_n(e^{\text{val} \cdot e \cdot s}_x, F))),$$

are now shown by induction on n as before, with inner inductions on the structure of F. □

Corollary 4.10 *Let ϕ be a Λ_n formula with free variables among $v_0, \ldots, v_{k-1}$. Then* $\text{HAP} \vdash \forall e(\text{Sat}_n(e, \ulcorner \phi \urcorner) \leftrightarrow \phi(e \cdot \ulcorner v_0 \urcorner, \ldots, e \cdot \ulcorner v_{k-1} \urcorner))$.

Proof Outer induction on n, inner induction on ϕ. □

5 Fixpoint theories

We return now to our main item of study: the intuitionistic theory of fixpoints over Heyting arithmetic, $\widehat{\text{ID}}^{\text{i}}_1$. Similar to the way HA is included in HAP we want to embed $\widehat{\text{ID}}^{\text{i}}_1$ in a corresponding theory in the logic of partial terms, extending the notion of strictly positive fixpoints to HAP. We denote this theory as $\widehat{\text{ID}}^{\text{i}}_1\text{P}$. We start by formally introducing these fixpoint theories.

5.1 $\widehat{\mathrm{ID}}_1^i$ and $\widehat{\mathrm{ID}}_1^i \mathrm{P}$

Let $\mathcal{L}$ be a fixed language extending $\mathcal{L}_{\mathrm{HA}}$. Let $P = (P_n)_{n\in\mathbb{N}}$ be a sequence of new predicate symbols such that P_n has arity n. Let $\mathcal{L}(P_n)$ be the language $\mathcal{L}$ with P_n added and $\mathcal{L}^P$ that of $\mathcal{L}$ with all P_n added. We will by $\mathrm{POS}_{P_n}(\mathcal{L})$ denote the set of formulae in $\mathcal{L}(P_n)$ whose free variables are *exactly* $\mathsf{v}_0, \ldots, \mathsf{v}_{n-1}$ and such that P_n occurs at least once in the formula and in strictly positive position only. That is, P_n may not occur in the antecedent of any implication. Let $\mathrm{POS}_P(\mathcal{L}) = \bigcup_{n\in\mathbb{N}} \mathrm{POS}_{P_n}(\mathcal{L})$. Note that this definition ensures that $\mathrm{POS}_P(\mathcal{L})$ is a union of disjoint classes, so exactly one of the P_ns occur in any formula of $\mathrm{POS}_P(\mathcal{L})$; the requirement that P_n actually occur is no real restriction on the $\mathrm{POS}_{P_n}(\mathcal{L})$. Formulae of $\mathrm{POS}_P(\mathcal{L})$ will be used as operators with P_n as parameters to generate (axioms for) strictly positive fixpoints, the *strictly positive operator forms (in $\mathcal{L}$)* from the introduction. Now we can formally define $\widehat{\mathrm{ID}}_1^i$ as follows: For each $\Phi \in \mathrm{POS}_{P_n}(\mathcal{L})$ we introduce a new n-ary predicate I_Φ and the axiom

$$\forall \vec{x}(\mathrm{I}_\Phi(\vec{x}) \leftrightarrow \Phi(\mathrm{I}_\Phi; \vec{x})). \tag{Ax$_\Phi$}$$

Similarly to above, we will by $\mathcal{L}(\mathrm{I}_\Phi)$ denote the language $\mathcal{L}$ extended with the (specific) symbol I_Φ and by $\mathcal{L}^{\mathrm{ID}}$ that of $\mathcal{L}$ with I_Φ added for every $\Phi \in \mathrm{POS}_P(\mathcal{L})$. Then $\widehat{\mathrm{ID}}_1^i(\Gamma)$ is $\mathrm{HA}(\mathcal{L}^{\mathrm{ID}})$ plus all (Ax$_\Phi$) for $\Phi \in \mathrm{POS}_P(\mathcal{L}) \cap \Gamma$, where Γ is some class of $\mathcal{L}^P$-formulae. $\widehat{\mathrm{ID}}_1(\Gamma)$ is $\widehat{\mathrm{ID}}_1^i(\Gamma) + \mathrm{LEM}$ (or equivalently, $\mathrm{PA}(\mathcal{L}^{\mathrm{ID}}) + \{\mathrm{Ax}_\Phi \mid \Phi \in \mathrm{POS}_P(\mathcal{L}) \cap \Gamma\}$). Often we will use this for classes Γ whose exact definition depends on the ambient language (like $\widehat{\mathrm{ID}}_1(\Pi_2)$ in the introduction), whence it is tacitly understood that the definition used for Γ is the one appropriate for $\mathcal{L}^P$ (i.e. Π_2 here means all Π_2-formulae in the language $\mathcal{L}^P$); if there is reason to avoid confusion we will write $\Gamma(\mathcal{L}^P)$ for this set. In case Γ is all of $\mathcal{L}^P$, and/or the language $\mathcal{L}$ is evident, we will often drop them from the notations. Thus $\widehat{\mathrm{ID}}_1^i$ is $\widehat{\mathrm{ID}}_1^i(\mathcal{L}_{\mathrm{HA}}^P)$.

We now extend this definition to fixpoint theories over HAP.

Definition 5.1 The theory $\widehat{\mathrm{ID}}_1^i \mathrm{P}$ is the LPT-theory in the language $\mathcal{L}_{\mathrm{HAP}}^{\mathrm{ID}}$ axiomatised by

1. the axioms of $\mathrm{HAP}(\mathcal{L}_{\mathrm{HAP}}^{\mathrm{ID}})$,
2. the schema (Ax$_\Phi$) for all $\Phi \in \mathrm{POS}_P(\mathcal{L}_{\mathrm{HAP}})$.

The theory $\widehat{\mathrm{ID}}_1^i \mathrm{P}(\Gamma)$, where Γ is a subset of $\mathcal{L}_{\mathrm{HAP}}^P$, is axiomatised by clause 1 above, and

3. the schema (Ax$_\Phi$) for all $\Phi \in \mathrm{POS}_P(\mathcal{L}_{\mathrm{HAP}}) \cap \Gamma$.

It follows from these definitions that the language of $\widehat{\mathrm{ID}}_1^i$ is contained in that of $\widehat{\mathrm{ID}}_1^i \mathrm{P}$; $\mathcal{L}_{\mathrm{HA}}^{\mathrm{ID}} \subset \mathcal{L}_{\mathrm{HAP}}^{\mathrm{ID}}$. Given Lemma 3.6, we can in fact make a stronger claim.

Lemma 5.2 $\widehat{\mathrm{ID}}_1^i \mathrm{P}$ *proves every theorem of* $\widehat{\mathrm{ID}}_1^i$.

Proof By Lemma 3.6 $\mathrm{HAP}(\mathcal{L}_{\mathrm{HAP}}^{\mathrm{ID}}) \vdash \mathrm{HA}(\mathcal{L}_{\mathrm{HA}}^{\mathrm{ID}})$. Hence by the same lemma all we need to check are the fixpoint axioms of $\widehat{\mathrm{ID}}_1^i$. Let $\mathrm{I}_\Phi(\vec{\mathsf{v}}) \leftrightarrow \Phi(\mathrm{I}_\Phi; \vec{\mathsf{v}})$ be such an axiom. Since this is literally an axiom of $\widehat{\mathrm{ID}}_1^i \mathrm{P}$ we are done. $\qquad\square$

The corresponding lemma holds for $\widehat{\mathrm{ID}}_1^i(\Gamma)$ and $\widehat{\mathrm{ID}}_1^i P(\Gamma)$ if Γ is a set of formulae (but recall that e.g. Λ and Λ_n below denote different sets in $\mathcal{L}_{\mathrm{HA}}^P$ and $\mathcal{L}_{\mathrm{HAP}}^P$).

5.2 Partial realizability interpretation of $\widehat{\mathrm{ID}}_1^i P$

The notion of realizability we are employing (Sect. 3.2) syntactically transforms a formula into a negative formula (Lemma 3.15). This transformation also preserves strictly positive occurrences of symbols and subformulae in many important instances. In particular, for $\Phi \in \mathrm{POS}_{P_n}(\mathcal{L})$, a strictly positive operator form in the n-ary predicate P_n, its 'realization' $x \underline{\mathrm{r}} \Phi$ will be in $\mathrm{POS}_{\underline{\mathrm{r}} P_n}(\mathcal{L})$, i.e. a strictly positive operator form in the $(n+1)$-ary predicate $\underline{\mathrm{r}} P_n$. Since a fixpoint axiom of $\widehat{\mathrm{ID}}_1^i P$ will be realized if we can provide functions transforming realizers of the fixpoint predicate into realizers of the corresponding operator applied to that predicate and vice versa, if we postulate these functions to be identity we get that $\mathrm{I}_\Phi(\vec{\mathrm{v}}) \leftrightarrow \Phi(\mathrm{I}_\Phi; \vec{\mathrm{v}})$ has a realizer provided that

$$\mathrm{v}_n \underline{\mathrm{r}} \, \mathrm{I}_\Phi(\vec{\mathrm{v}}) \leftrightarrow \mathrm{v}_n \underline{\mathrm{r}} \, \Phi(\mathrm{I}_\Phi; \vec{\mathrm{v}}) \tag{2}$$

holds. By above this biconditional is itself (equivalent to) a strictly positive fixpoint axiom, and since realizability is a transformation into negative formulae, (2) is actually equivalent to an axiom of $\widehat{\mathrm{ID}}_1^i P(\Lambda)$. In this section we make this realizability interpretation of $\widehat{\mathrm{ID}}_1^i P$ in $\widehat{\mathrm{ID}}_1^i P(\Lambda)$ precise (Theorem 5.6).

First, we augment the realizability transformation with a clause for the 'fixpoint generators' P, and subsequently for all fixpoint predicates I_Φ, in the manner of [5, §6].

Definition 5.3 For the language $\mathcal{L}_{\mathrm{HAP}}^P$, we stipulate $\underline{\mathrm{r}} P_n = P_{n+1}$, that is

$$\tau \underline{\mathrm{r}} \, P_n(\vec{x}) \text{ is } P_{n+1}(\vec{x}, \tau)$$

in the final clause of the definition of realizability.

Lemma 5.4 *In $\mathcal{L}_{\mathrm{HAP}}^P$ with realizability as per Definition 5.3 we have that $\mathrm{v}_n \underline{\mathrm{r}} \, \Phi$ for $\Phi \in \mathrm{POS}_{P_n}(\mathcal{L}_{\mathrm{HAP}})$ is in $\mathrm{POS}_{P_{n+1}}(\mathcal{L}_{\mathrm{HAP}}) \cap \Lambda$. More generally, let $\mathcal{L}$ extend $\mathcal{L}_{\mathrm{HAP}}$ with a set of relation symbols, R being among them. Suppose $\underline{\mathrm{r}} R$ is not the same as $\underline{\mathrm{r}} Q$ for any other relation Q. If $\phi \in \mathcal{L}$ has only strictly positive occurrences of R and τ is a term, $\tau \underline{\mathrm{r}} \phi$ is negative and has only strictly positive occurrences of $\underline{\mathrm{r}} R$, the number of which equals the number of occurrences of R in ϕ, and $\mathrm{FV}(\tau \underline{\mathrm{r}} \phi) = \mathrm{FV}(\phi) \cup \mathrm{FV}(\tau)$ if R occurs in ϕ.*

Proof The statement about negativity follows from Lemma 3.15. The rest is shown by induction on complexity of ϕ.

- The atomic cases are trivial.
- The inductive steps for the Boolean connectives are straightforward, so we present only the argument for the implication.

Thus consider the case when ϕ is an implication $\psi \to \theta$, where ψ does not contain R, θ has only strictly positive occurrences of R and for all terms σ_1 and σ_2 do we have that $\sigma_1 \underline{r} \psi$ has no occurrence of $\underline{r}R$, the number of occurrences of $\underline{r}R$ in $\sigma_2 \underline{r} \theta$ is the same as the number of occurrences of R in θ, and if θ has at least one occurrence of R then the free variables of $\sigma_2 \underline{r} \theta$ are those of θ and σ_2.

Let τ be a term. Then

$$\tau \underline{r} (\psi \to \theta) = \forall x (x \underline{r} \psi \to (\tau x{\downarrow} \wedge \tau x \underline{r} \theta))$$

has only strictly positive occurrences of $\underline{r}R$ by induction hypothesis, since all of these are in $\tau x \underline{r} \theta$. Thus the number of such occurrences equals the number of such occurrences in $\tau x \underline{r} \theta$, which by induction hypothesis is the number of occurrences of R in θ, i.e. in $\psi \to \theta$. If R occurs in $\psi \to \theta$ then it occurs in θ, whence

$$\begin{aligned}
\mathrm{FV}(\tau \underline{r} (\psi \to \theta)) &= \mathrm{FV}(\forall x (x \underline{r} \psi \to (\tau x{\downarrow} \wedge \tau x \underline{r} \theta))) \\
&= (\mathrm{FV}(\psi) \cup ((\mathrm{FV}(\tau) \cup \mathrm{FV}(\theta)) \cup \{x\})) \backslash \{x\} \\
&= \mathrm{FV}(\psi) \cup \mathrm{FV}(\theta) \cup \mathrm{FV}(\tau) \\
&= \mathrm{FV}(\psi \to \theta) \cup \mathrm{FV}(\tau)
\end{aligned}$$

by Lemma 3.15 and induction hypothesis, since x does not occur in ψ, θ and τ.

- Since the existential case contains all essential ideas from the universal case but is slightly more involved, we omit the universal case.

  Consider the case when ϕ starts with an existential quantifier, $\exists x \psi$, where ψ only has strictly positive occurrences of R and for any term σ and formula θ with only strictly positive occurrences of R and the same complexity as ψ, $\sigma \underline{r} \theta$ is strictly positive in $\underline{r}R$, has as many occurrences of $\underline{r}R$ as θ has of R and satisfies $\mathrm{FV}(\sigma \underline{r} \theta) = \mathrm{FV}(\theta) \cup \mathrm{FV}(\sigma)$ if R occurs in θ.

  Let τ be a term. By convention we can assume its variables are not bound in $\exists x \psi$. Then

$$\tau \underline{r} (\exists x \psi) = \mathbf{p}_r \tau \underline{r} (\psi(x/\mathbf{p}_l \tau))$$

Since $\psi(x/\mathbf{p}_l \tau)$ has the same complexity as ψ the induction hypothesis applies, whence $\mathbf{p}_r \tau \underline{r} (\psi(x/\mathbf{p}_l \tau))$ is strictly positive in $\underline{r}R$ and has as many occurrences of $\underline{r}R$ as $\psi(x/\mathbf{p}_l \tau)$, and hence $\exists x \psi$, has of R. Also, if R occurs in $\exists x \psi$ it occurs in $\psi(x/\mathbf{p}_l \tau)$ whence

$$\begin{aligned}
\mathrm{FV}(\tau \underline{r} \exists x \psi) &= \mathrm{FV}(\mathbf{p}_r \tau \underline{r} (\psi(x/\mathbf{p}_l \tau))) \\
&= \mathrm{FV}(\psi(x/\mathbf{p}_l \tau)) \cup \mathrm{FV}(\mathbf{p}_r \tau) \\
&= \mathrm{FV}(\psi) \backslash \{x\} \cup \mathrm{FV}(\tau) \\
&= \mathrm{FV}(\exists x \psi) \cup \mathrm{FV}(\tau)
\end{aligned}$$

by induction hypothesis.

This concludes the proof. □

Another consequence of the above is that $x \mathrel{\underline{r}} \Phi(I_\Phi; \vec{y})$ is strictly positive in $\underline{r} I_\Phi$, the particular choice of which we can now specify.

Definition 5.5 For the language $\mathcal{L}^{ID}_{HAP}$, given $\Phi \in POS_{P_n}(\mathcal{L}_{HAP})$ we stipulate $\underline{r} I_\Phi = I_{v_n \underline{r} \Phi}$, the almost negative strictly positive operator form from Lemma 5.4, that is

$$\tau \mathrel{\underline{r}} I_\Phi(\vec{x}) \text{ is } I_{v_n \underline{r} \Phi}(\vec{x}, \tau)$$

in the final clause of the definition of realizability.

Theorem 5.6 *For the realizability interpretation given by Definitions 3.14 and 5.5,* $\widehat{ID}^i_1 P \vdash \varphi \Rightarrow \widehat{ID}^i_1 P(\Lambda) \vdash \exists x (x \mathrel{\underline{r}} \varphi)$ *for* $\varphi \in \mathcal{L}^{ID}_{HAP}$ *and* $x \notin FV(\varphi)$.

Proof By Lemma 3.18 and compactness it is enough to show that for each non-logical axiom φ of $\widehat{ID}^i_1 P$ there is a combinator τ with $\widehat{ID}^i_1 P(\Lambda) \vdash \tau \downarrow \wedge \tau \mathrel{\underline{r}} \varphi$. Now, every axiom except the induction and fixpoint schemata, HAP2 and the third part of HAP5 is an equation true in $HAP(\mathcal{L}^{ID}_{HAP})$, and hence realised by 0 there; HAP2 and the third part of HAP5 are realized by **id**. As for the induction axiom Ind_φ,

$$HAP(\mathcal{L}^{ID}_{HAP}) \vdash \mathbf{r} \mathrel{\underline{r}} Ind_\varphi$$

For the fixpoint axioms, we observe the following. By Definition 5.5 realizability is a transformation of the language $\mathcal{L}^{ID}_{HAP}$ into the fragment defined by the almost negative strictly positive operator forms. Thus given a fixpoint axiom $\forall \vec{y}(I_\Phi(\vec{y}) \leftrightarrow \Phi(I_\Phi; \vec{y}))$ with $\Phi \in POS_{P_n}(\mathcal{L}_{HAP})$ let $\Psi \in POS_{P_{n+1}}(\mathcal{L}_{HAP})$ be the operator form from Lemma 5.4 such that $\underline{r} I_\Phi = I_\Psi$. Then

$$
\begin{aligned}
\widehat{ID}^i_1 P(\Lambda) \vdash x \mathrel{\underline{r}} I_\Phi(\vec{y}) &\leftrightarrow I_\Psi(\vec{y}, x) && \text{definition} \\
&\leftrightarrow \Psi(I_\Psi; \vec{y}, x) && \text{fixpoint axiom} \\
&\leftrightarrow \Psi(\vec{y}, x)(I_\Psi) && \text{definition} \\
&\leftrightarrow (v_n \mathrel{\underline{r}} \Phi)(\vec{y}, x)(I_\Psi) && \text{Lemma 5.4} \\
&\leftrightarrow (x \mathrel{\underline{r}} \Phi(\vec{y}))(I_\Psi) && \text{Lemma 3.16} \\
&\leftrightarrow (x \mathrel{\underline{r}} \Phi(\vec{y}))(\underline{r} I_\Phi) && \text{definition}
\end{aligned}
$$

Appealing to Lemma 3.17 this equivalence, with x universally quantified, is equivalent to $\mathbf{p} \cdot \mathbf{id} \cdot \mathbf{id} \mathrel{\underline{r}} (I_\Phi(\vec{y}) \leftrightarrow \Phi(I_\Phi; \vec{y}))$. Hence

$$\widehat{ID}^i_1 P(\Lambda) \vdash (\lambda \vec{y}.\mathbf{p} \cdot \mathbf{id} \cdot \mathbf{id}) \mathrel{\underline{r}} \forall \vec{y}(I_\Phi(\vec{y}) \leftrightarrow \Phi(I_\Phi; \vec{y})),$$

which concludes the proof.   □

We close the section by proving that $\widehat{ID}^i_1 P(\Lambda)$ is a conservative extension of HAP. The proof utilises the partial satisfaction predicates introduced in Sect. 4.

Theorem 5.7 $\widehat{\mathrm{ID}}_1^{\mathrm{i}}\mathrm{P}(\Lambda)$ *is interpretable in* HAP, *keeping* $\mathcal{L}_{\mathrm{HAP}}$ *fixed.*

Proof We define a compositional interpretation $\mathcal{F}\colon \mathcal{L}_{\mathrm{HAP}}^{\mathrm{ID}} \longrightarrow \mathcal{L}_{\mathrm{HAP}}$ which fixes $\mathcal{L}_{\mathrm{HAP}}$. To that extent, for $k \in \mathbb{N}$ let T_k be the term $\mathbf{id}_{\ulcorner \vec{\mathsf{v}}_{<k} \urcorner}^{\vec{\mathsf{v}}_{<k}}$ (the identity combinator modified so that $\ulcorner \mathsf{v}_j \urcorner$ is mapped to (the value of) v_j), so that T_k has exactly $\mathsf{v}_0, \ldots, \mathsf{v}_{k-1}$ free, $\mathrm{HAP} \vdash T_k{\downarrow}$ and $\mathrm{HAP} \vdash T_k \cdot \ulcorner \mathsf{v}_j \urcorner = \mathsf{v}_j$ for all $j < k$. Take an almost negative $\Phi \in \mathrm{POS}_{P_k}(\mathcal{L}_{\mathrm{HAP}})$ and let n be the smallest index for which $\Phi \in \Lambda_n$, by Lemma 4.2. Set $\phi = \Phi(\lambda\vec{\mathsf{v}}_{<k}.\mathsf{Sat}_n(T_k, \mathsf{v}_k); \vec{\mathsf{v}}_{<k})$. By Theorem 4.9 $\phi \in \mathcal{L}_{\mathrm{HAP}}$, and by Lemma 4.5 $\phi \in \Lambda_n$. Diagonalisation (Lemma 4.6) yields a formula $\psi \in \Lambda_n$ with free variables $\mathsf{v}_0, \ldots, \mathsf{v}_{k-1}$ and

$$\mathrm{HAP} \vdash \psi(\vec{x}) \leftrightarrow \phi(\vec{x}, \ulcorner \psi \urcorner).$$

Defining $\mathcal{F}(\mathrm{I}_\Phi) = \psi$ we thus see

$$\begin{aligned}
\mathrm{HAP} \vdash \mathcal{F}(\mathrm{I}_\Phi(\vec{x})) &\leftrightarrow \psi(\vec{x}) \\
&\leftrightarrow \phi(\vec{x}, \ulcorner \psi \urcorner) \\
&\leftrightarrow \Phi(\lambda\vec{\mathsf{v}}_{<k}.\mathsf{Sat}_n(T_k, \ulcorner \psi \urcorner); \vec{x}) \\
&\leftrightarrow \Phi(\lambda\vec{\mathsf{v}}_{<k}.\psi(\vec{\mathsf{v}}_{<k}); \vec{x}) \\
&\leftrightarrow \Phi(\psi; \vec{x}) \\
&\leftrightarrow \Phi(\mathcal{F}(\mathrm{I}_\Phi); \vec{x}) \\
&\leftrightarrow \mathcal{F}(\Phi(\mathrm{I}_\Phi; \vec{x}))
\end{aligned}$$

by Corollary 4.10. Setting $\mathcal{F}(\Psi) = \bot$ for $\Psi \in \mathrm{POS}_P(\mathcal{L}_{\mathrm{HAP}})\backslash\Lambda$ (since these are inconsequential), the interpretation of the remaining axioms of $\widehat{\mathrm{ID}}_1^{\mathrm{i}}\mathrm{P}(\Lambda)$ are immediate. Consequently $\mathrm{HAP} \vdash \mathcal{F}[\widehat{\mathrm{ID}}_1^{\mathrm{i}}\mathrm{P}(\Lambda)]$.   □

6 Conclusion

Tying together the above results, we can prove the wanted conservativity result.

Theorem 6.1 $\widehat{\mathrm{ID}}_1^{\mathrm{i}}$ *is conservative over* HA.

Proof Let φ be a sentence of $\mathcal{L}_{\mathrm{HA}}$. Then we have the following chain of implications:

$$\begin{aligned}
\widehat{\mathrm{ID}}_1^{\mathrm{i}} \vdash \varphi &\Rightarrow \widehat{\mathrm{ID}}_1^{\mathrm{i}}\mathrm{P} \vdash \varphi && \text{Lemma 5.2} \\
&\Rightarrow \widehat{\mathrm{ID}}_1^{\mathrm{i}}\mathrm{P}(\Lambda) \vdash \exists x (x \underline{\mathrm{r}}\, \varphi) && \text{Theorem 5.6} \\
&\Rightarrow \mathrm{HAP} \vdash \exists x (x \underline{\mathrm{r}}\, \varphi) && \text{Theorem 5.7} \\
&\Rightarrow \mathrm{HA} \vdash \varphi && \text{Proposition 3.20} \qquad □
\end{aligned}$$

The argument we have presented combines two interpretations: a realizability interpretation of a fixpoint theory into the subtheory of fixpoints for almost negative operator forms and a direct interpretation of this subtheory in Heyting arithmetic. Full

conservativity does not directly follow from these reductions because realizability and provability do not coincide for Heyting arithmetic: there are statements in the language of arithmetic which are (Kleene-)realizable yet not provable. As the two notions agree on the fragment of almost negative formulae, conservativity for this class of sentences holds. But by lifting the two interpretations to the corresponding theories formulated over Beeson's richer logic of partial terms the stronger conservation result becomes a consequence of van den Berg and van Slooten's theorem; for arithmetic formulae, provability in Heyting arithmetic subsumes realizability in Heyting arithmetic with partial terms.

A key ingredient in our argument is the existence of partial satisfaction predicates for a formula-hierarchy which exhausts the almost negative formulae. These partial satisfaction predicates do not rely on the presence of partial terms in any essential way, and arithmetic counterparts are available which, provably in HA, are compositional satisfaction predicates for the arithmetic formulae at the same level. A natural question is whether there exist partial satisfaction predicates for classes of formulae including the positive connectives $\vee$ and $\exists$, whether in the language with partial terms or pure arithmetic. A partial negative answer to this question was provided by Leivant in [15], who, improving a result of de Jongh [16], gave lower bounds on the negative nesting depth of possible satisfaction predicates for formula classes which include all propositional combinations of Σ_1-, Π_1- or Σ_2-formulae, to some fixed nesting depth. Whether, more generally, there is a stratification of the entire language which admits partial satisfaction predicates in HA is, to the authors' knowledge, unknown, which is somewhat surprising given the importance of such predicates in the proof theory of classical systems of arithmetic.

There can, however, not be a hierarchy of arithmetic formulae which (i) is closed under substitution in strictly positive positions (Lemma 4.5), (ii) is closed under diagonalisation (Lemma 4.6), (iii) exhausts the *entire* language $\mathcal{L}_{HA}$, and (iv) has partial satisfaction predicates for every level. Such a hierarchy with corresponding satisfaction predicates would give rise to a direct interpretation of $\widehat{ID}_1^i$ in HA by generalising Theorem 5.7, but it would also allow the classical theory $\widehat{ID}_1$ to be interpreted in PA by the same argument. This cannot be the case because $\widehat{ID}_1$ is strictly stronger than PA. On the other hand, this line of reasoning shows that $\widehat{ID}_1(\Lambda)$ *is* conservative over PA, again by generalising Theorem 5.7. Thus, the *strictly* positive almost negative operator forms gives a weaker classical fixpoint theory than the merely positive almost negative ones, since the latter is equivalent to $\widehat{ID}_1$. Attempting to carry out our argument for this class of fixpoints fails on Lemma 4.5, since Λ_n is not closed under substitutions into merely positive positions. A final note is that, since Π_1 has a Π_1 satisfaction predicate and is closed under substitutions into strictly positive positions, $\widehat{ID}_1(\Pi_1)$ is conservative over PA essentially by replacing Λ_n by Π_1 in the proof of Theorem 5.7. On the other hand $\widehat{ID}_1(\Pi_2)$ is not conservative over PA, since it interprets the Kripke-Feferman theory of truth KF (Feferman, [17]) which proves the consistency of PA. It would therefore be interesting to identify the subclasses Γ of Π_2 forms for which $\widehat{ID}_1(\Gamma)$ is still conservative.

We do believe this argument generalises to finite iterations $\widehat{ID}_n^i$ of fixpoint theories, however. The main idea would then be to repeat the application of Theorem 5.7 inductively to reduce $\widehat{ID}_{n+1}^i P(\Lambda)$ to $\widehat{ID}_n^i P(\Lambda)$, viewing HAP as $\widehat{ID}_0^i P(\Lambda)$ ($\widehat{ID}_n^i P(\Lambda)$ should

mean *all* operator forms are almost negative, not just the nth 'top layer'). The possible issue we see in Sect. 4.2 is that Theorem 4.9 requires (primitive) recursive transformations from Λ_n to Λ_n^{nf} as in Lemma 4.4, which is not as straightforward for countable vocabularies. This should be manageable by more careful attention to coding, or finally by a compactness argument in Theorem 5.7. The proof of Theorem 5.7 should otherwise generalise, essentially by placing an '$\mathcal{F}$' defined by induction hypothesis in front of every appearance of Φ except the first. In the rest of the argument of Theorem 6.1, generalising in this direction would add a countable number of relation symbols and axioms governing these, and Lemma 5.2 clearly generalises to this case. For the results leading up to Theorem 5.6, prominently Lemma 5.4, we have taken some care not to specify the relation symbols in the base language, so reformulating Theorem 5.6 slightly this step should also generalise by induction. Finally, the final step would remain unchanged.

A natural question would be whether a generalisation of this argument works also for some transfinite iterations $\widehat{\mathrm{ID}}_\alpha^{\mathrm{i}}$. While we have no clear ideas in this direction, it's a reasonable question for future work.

References

1. Buchholz, W.: An intuitionistic fixed point theory. Arch. Math. Log. **37**, 21–27 (1997). https://doi.org/10.1007/s001530050079
2. Arai, T.: Some results on cut-elimination, provable well-orderings, induction and reflection. Ann. Pure Appl. Log. **95**, 93–184 (1998). https://doi.org/10.1016/S0168-0072(98)00020-7
3. Rüede, C., Strahm, T.: Intuitionistic fixed point theories for strictly positive operators. Math. Log. Q. **48**, 195–202 (2002). https://doi.org/10.1002/1521-3870(200202)48:2<195::AID-MALQ195>3.0.CO;2-S
4. Arai, T.: Quick cut-elimination for strictly positive cuts. Ann. Pure Appl. Log. **162**, 807–815 (2011). https://doi.org/10.1016/j.apal.2011.03.002
5. Buchholz, W.: The $\Omega_{\mu+1}$-rule. In: Buchholz, W., Feferman, S., Pohlers, W., Sieg, W. (eds.) Iterated Inductive Definitions and Subsystems of Analysis: Recent Proof-Theoretical Studies, vol. 897, pp. 188–233. Springer, Berlin (1981)

6. Beeson, M.J.: Foundations of Constructive Mathematics: Metamathematical Studies. Springer, Berlin (1985)
7. van den Berg, B., van Slooten, L.: Arithmetical conservation results. Indag. Math. **29**, 260–275 (2018). https://doi.org/10.1016/j.indag.2017.07.009
8. Troelstra, A.S., van Dalen, D.: Constructivism in Mathematics: An Introduction, vol. I. Elsevier, Amsterdam (1988)
9. Troelstra, A.S.: Realizability. In: Buss, S.R. (ed.) Handbook of Proof Theory, vol. 137, pp. 407–473. Elsevier, Amsterdam (1998)
10. van Slooten, L.: Arithmetical conservativity results, a theory of operations and Goodman's theorem. Master's thesis, Universiteit Utrecht (2014)
11. Troelstra, A.S., van Dalen, D.: Constructivism in Mathematics: An Introduction, vol. II. Elsevier, Amsterdam (1988)
12. Burr, W.: Fragments of Heyting arithmetic. J. Symb. Log. **65**, 1223–1240 (2000). https://doi.org/10.2307/2586698
13. Burr, W.: The intuitionistic arithmetical hierarchy. In: van Eijk, J., van Oostrom, V., Visser, A. (eds.) Logic Colloquium '99: Proceedings of the Annual European Summer Meeting of the Association for Symbolic Logic, Held in Utrecht, Netherlands, August 1–6, 1999, vol. 17, pp. 51–59. A K Peters, Ltd., Natick (2004)
14. Hájek, P., Pudlák, P.: Metamathematics of First Order Arithmetic. Springer, Berlin (1993)
15. Leivant, D.: Implicational complexity in intuitionistic arithmetic. J. Symb. Log. **46**, 240–248 (1981). https://doi.org/10.2307/2273617
16. de Jongh, D.H.J.: Formulas of one propositional variable in intuitionistic arithmetic. Technical Report 73-03, Department of Mathematics, University of Amsterdam, Amsterdam (1973). Reprinted in: Troelstra, A. S., van Dalen, D. (eds.) The L. E. J. Brouwer Centenary Symposium, vol. 110, pp. 51–64. North-Holland Publishing Company, Amsterdam (1982)
17. Feferman, S.: Reflecting on incompleteness. J. Symb. Log. **56**, 1–49 (1991). https://doi.org/10.2307/2274902