Basic Research in Computer Science

BRICS RS-03-12 Hernest & Kohlenbach: A Complexity Analysis of Functional Interpretations

A Complexity Analysis of Functional Interpretations

Mircea-Dan Hernest
Ulrich Kohlenbach

BRICS Report Series **RS-03-12**

ISSN 0909-0878 **February 2003**

**Copyright © 2003, Mircea-Dan Hernest & Ulrich Kohlenbach.
BRICS, Department of Computer Science
University of Aarhus. All rights reserved.**

**Reproduction of all or part of this work
is permitted for educational or research use
on condition that this copyright notice is
included in any copy.**

**See back inner page for a list of recent BRICS Report Series publications.
Copies may be obtained by contacting:**

**BRICS
Department of Computer Science
University of Aarhus
Ny Munkegade, building 540
DK–8000 Aarhus C
Denmark
Telephone: +45 8942 3360
Telefax: +45 8942 3255
Internet: BRICS@brics.dk**

BRICS publications are in general accessible through the World Wide Web and anonymous FTP through these URLs:

`http://www.brics.dk`
`ftp://ftp.brics.dk`
This document in subdirectory `RS/03/12/`

A complexity analysis of functional interpretations

Mircea-Dan Hernest* and Ulrich Kohlenbach†

February, 2003

Abstract

We give a quantitative analysis of Gödel's functional interpretation and its monotone variant. The two have been used for the extraction of programs and numerical bounds as well as for conservation results. They apply both to (semi-)intuitionistic as well as (combined with negative translation) classical proofs. The proofs may be formalized in systems ranging from weak base systems to arithmetic and analysis (and numerous fragments of these). We give upper bounds in basic proof data on the depth, size, maximal type degree and maximal type arity of the extracted terms as well as on the depth of the verifying proof. In all cases terms of size linear in the size of the proof at input can be extracted and the corresponding extraction algorithms have cubic worst-time complexity. The verifying proofs have depth linear in the depth of the proof at input and the maximal size of a formula of this proof.

*GKLI, Mathematisches Institut der Universität München, Theresienstr. 39, D-80333 München, Germany, danher@mathematik.uni-muenchen.de; the first half of the work involved was carried at BRICS, Department of Computer Science, University of Aarhus; the second half of the work was supported by the "Deutsche Forschungsgemeinschaft" via the "Graduiertenkolleg Logik in der Informatik" (GKLI)

†BRICS, Department of Computer Science, University of Aarhus, Ny Munkegade 540, DK-8000 Aarhus C, Denmark, kohlenb@brics.dk; partly supported by the Danish Natural Science Research Council (Grant no. 21-02-0474)

Contents

1 Introduction

This paper investigates the complexity of the extraction algorithms for effective data (such as programs and bounds) from proofs provided by Gödel's functional ('Dialectica') interpretation and its monotone variant. The subject of extracting programs from proofs already has a long history. The techniques used can be roughly divided in two categories according to whether they are based on cut-elimination, normalization and related methods or on so-called proof interpretations. The latter typically make use of functionals

of higher type. Prominent proof interpretations are realizability interpretations, particularly Kreisel's [41] modified realizability (see [52] for a survey) and Gödel's functional interpretation (first published in [21], see [3] for a survey). The no-counterexample interpretation (*n.c.i.*) due to Kreisel [39, 40] is sometimes viewed as a simplification of the functional interpretation (it uses only types of degree ≤ 2). In fact *n.c.i.* is not a real alternative since it has a bad behavior with respect to the modus ponens rule `MP`. This is overcome only if `MP` is interpreted by functional interpretation (see [34]).

Cut-elimination, normalization and the related ε-substitution method globally rebuild the given proof thereby increasing its length in a potentially non-elementary recursive way. In contrast, proof interpretations extract witnessing terms by recursion on the given proof tree which remains essentially unchanged in its structure. The latter techniques consequently enjoy full modularity: the global realizers of a proof can be computed from realizers of lemmas used in the proof. This suggests a radically lower complexity of the procedure and a radically smaller size of the extracted programs. Even though the latter would not be in normal form[1] they can be used substantially in many ways without having to normalize them. One merely exploits properties which can be established inductively over their structure with the use of logical relations (like, e.g., Howard's [24] notion of majorizability).

Both (modified) realizability and functional interpretations are applicable to a vast variety of formal systems and provide characterizations of their provably total programs. They had originally been applied to arithmetic in all finite types. They were subsequently adapted to various fragments thereof all the way down to weak systems of bounded arithmetic [10, 33, 45] or – more recently – the poly-time arithmetic of [5, 48]. They were extended to analysis [16, 42, 50], type theories [19] and fragments of set theory [8].

Realizability and functional interpretations cannot be directly applied to classical systems. A canonical manner of interpreting classical proofs would be by first translating them to intuitionistic proofs via a so-called negative translation and subsequently applying intuitionistic proof interpretations. However this fails for (modified) realizability since it extracts empty programs from negative formulas. The problem can be partly overcome by using an additional intermediate interpretation, the so-called Friedman-Dragalin A-translation [14, 17] and its variants [11] [2]. Unlike realizability interpreta-

[1] Normalization would bring back the aforementioned complexities.

[2] See, e.g., [6, 44] for examples of program extractions using this approach. One draw-

tions, functional interpretations are sound for the so-called Markov principle and therefore feature extraction of programs from arbitrary proofs in fairly rich classical systems, like Peano arithmetic in all finite types $\mathtt{PA}^\omega$ (see also Section 5.3). Hence the need for an intermediate translation is avoided when using functional interpretations. Moreover, monotone functional interpretation can extract programs from proofs $\mathcal{T} \vdash \forall x^\rho \exists y^\tau Rec(x,y)$ in highly unconstructive systems $\mathcal{T}$ which contain, e.g., the binary König lemma. Here τ is an arbitrary finite type, ρ is a finite type of degree (aka level) at most 1 and $Rec(x,y)$ is a specification which must[3] be decidable if $\mathcal{T}$ is classical (we actually take it quantifier-free). This gives functional interpretations the ability of extracting programs and other effective data[4] under certain conditions from ineffective proofs (*proof mining*). Monotone-functional-interpretation-based proof mining has already produced important results in computational analysis and has helped to obtain new results in mathematical analysis (see, e.g., [26, 28, 29, 31, 32, 35, 36, 37, 38]).

A natural question that arises is whether such applications which were obtained 'by hand' could be automated or at least computer aided by implementing functional interpretations. In order to evaluate the feasibility of such a tool it is important to investigate the complexity aspects of functional interpretations. In the present paper we obtain upper bounds on the size of the terms which express the extracted programs. The interpretation algorithms only write down the extracted terms, proceeding by recursion on the input proof structure, see Section 3. It follows that their running time is proportional with the size of the extracted terms. Hence we obtain the time complexity of the extraction algorithms as a consequence of our quantitative analysis. Let n denote the size of the input proof $\mathcal{P}$ and m denote the maximal size of a formula of $\mathcal{P}$. Due to the modularity of functional interpretations, these algorithms feature an almost linear time complexity, namely $O(m^2 \cdot n)$ even for classical and analytical proofs. The *almost* refers to the fact that m is much smaller than n in most practical cases. In any case this time complexity is at most $O(n^3)$, a result previously obtained by Alexi in

back of this method is the limited modularity feature: only a restricted class of lemmas can be used in the proof at input. In contrast, functional-interpretations-based techniques feature full modularity: proofs at input may use arbitrary lemmas. See also [22] for applications of a form of recursive realizability.

[3]This restriction is generally unavoidable for classical proofs but is not necessary for intuitionistic proofs.

[4]Such as numerical bounds.

[1] for an ad-hoc program-extraction technique for intuitionistic proofs only. Since the design of Alexi's technique was driven by the optimal-time-overhead issue, *cubic* is probably the best worst-time-complexity one can expect from any program-extraction technique. We also give upper bounds on depths of the resulting verifying proofs – this is interesting for quantitative conservation results. In particular we obtain the feasibility of WKL–elimination for Π^0_2–sentences over primitive recursive arithmetic[5] in all finite types by means of syntactic translations. Our technique is immediately implementable and in addition provides a term extraction procedure from analytical proofs.

There exists a research line in extractive proof theory which is aimed at characterizing the classes of proofs from which programs belonging to certain complexity classes are extracted. Usually the *feasible* complexity classes are of interest, particularly poly-time, see for example [10, 48]. The issue of characterizing the complexity of provably total function(al)s of a theory is completely separate from the present paper's topic. We are here concerned with the performance of the extraction algorithm and not with the one of the extracted programs.

The monotone variant of Gödel's functional interpretation was developed by the second author in [26]. It takes into account that most applications of functional interpretation in recent years both to concrete proofs in numerical analysis and to conservation results do not actually use terms which realize the Gödel functional interpretation but terms which majorize[6] (some) realizers. Monotone functional interpretation extracts majorizing terms which are simpler than the actual realizers produced by functional interpretation. This is due to the much simpler treatment of $\mathtt{CT}\wedge$, see Proposition 3.22 and the paragraph following Definition 4.16. Also the treatment of induction axioms is much simpler, see Section 5. Moreover, the bound on verifying proof depth is better in the monotone case if the underlying logical system fairly supports monotone functional interpretation, see Remark 4.19.

1.1 Outline of the main results

We introduce the weak base system $\mathtt{EIL}^\omega$, a short for '(weakly extensional) extended intuitionistic equality logic in all finite types'. $\mathtt{EIL}^\omega$ contains only

[5]This was first shown in [2] for a second–order fragment with a formalized forcing technique.

[6]Majorization is understood in the sense of Howard [24] mentioned before.

the tools which are strictly necessary for carrying out the functional interpretation even for the most rudimentary intuitionistic systems.

We present upper bounds for the following quantitative measures of realizing/majorizing terms t extracted from proofs $\mathcal{P}$ in both semi-intuitionistic[7] and classical systems based on $\mathtt{EIL}^\omega$ up to the analytical system $\mathtt{PA}^\omega + \mathtt{AC}_0 + \mathtt{WKL}$:

- the maximal degree (arity) of a subterm of t, denoted mdg (mar);
- the depth of t, denoted d (assuming a tree representation of the terms);
- the size of t, denoted S (number of all constants and variables used to build t).

We also give upper bounds on the depth[8] of the verifying proof and time overhead of the extraction algorithm, here denoted ∂_v and θ respectively. For the extraction procedure we consider both the usual [21] and the monotone variant [26] of functional interpretation. We first consider a binary-tree representation for terms, see also Footnote 32. Such a representation is more intuitive and therefore provides a better exposition of the bounds for mdg, mar. However it turns out that the same extracted terms have smaller size if represented in a more economic manner using pointers[9], see Section 3.4. Since their definition does not depend on the term representation, the bounds for mdg and mar still hold. From Section 3.4 on it is tacitly assumed that terms are represented in the economic manner. A representation for types becomes necessary only at the moment that we are interested in the space/time overhead of the extraction algorithm, see Section 3.5.

Let us denote by ∂ the depth and by S_i, S_c, S_m the size (in the sense of Definition 3.33) of $\mathcal{P}$ and for a formula A by

- vdg (var) the maximal degree (arity) of a variable occurring in A;
- id (fd, ld) the implication (forall, logical) depth of A, namely the maximal number of $\rightarrow$ ($\forall$, all logical constants) on a path from root to leaves in the usual tree representation of A; by $fid :\equiv max\{fd, id\}$;

[7] Here *semi-intuitionistic* means intuitionistic plus a version of Markov's principle $\mathtt{MK}$ and independence of premises for universal premises $\mathtt{IP}_\forall$, see Section 3.1 for details.

[8] Proofs are represented as trees, see also the last paragraph of Section 1.2.

[9] It would be possible to extract other terms which have the same smaller size in binary-tree representation. However, the bounds for mdg, mar no longer hold in such a case, see also the remarks following Theorem 3.37.

- qs the number of all quantifiers (including[10] $\vee$) and free variables of A;
- ls the number of all $\forall, \exists, \wedge, \vee, \rightarrow, \bot, =$ and free variables of A.

We prove that (relative to our underlying deductive framework $\mathtt{EIL}^\omega$)

- mdg and mar do not depend on ∂; the difference between mdg (mar) and the maximal degree (arity) of a variable occurring in an axiom of $\mathcal{P}$ is linear (quadratic) in the maximal complexity of an axiom of $\mathcal{P}$;
- d is linear in the maximal complexity of an axiom of $\mathcal{P}$ and ∂;
- S is linear in the size of $\mathcal{P}$ (here we use the economic representation of terms); also exponential in logarithm of the maximal complexity of an axiom of $\mathcal{P}$ and ∂ (this holds for the binary-tree representation of terms as well);
- ∂_v is linear in ∂ and the maximal complexity of an axiom of $\mathcal{P}$.

More precisely, for **semi-intuitionistic** proofs $\mathcal{P}$ we have the following situation (below 'FI' means 'functional interpretation'):

	usual FI	monotone FI
mdg	$O(1) + vdg_{\mathrm{o}} + id_{\mathrm{o}}$	$O(1) + vdg_{\mathrm{o}} + id_{\mathrm{o}}$
mar	$O(1) + var_{\mathrm{o}} + qs_{\mathrm{o}} \cdot id_{\mathrm{o}}$	$O(1) + var_{\mathrm{o}} + qs_{\mathrm{o}} \cdot id_{\mathrm{o}}$
d	$O(ld_1) + qs_{\mathrm{o}} \cdot \partial$	$O(1) + qs_{\mathrm{o}} \cdot \partial$
S	$O(S_i)$, $O(ls_1 \cdot qs_{\mathrm{o}}^{\partial})$	$O(S_m)$, $O(qs_{\mathrm{o}}^{\partial})$
∂_v	$O(ld_1 + \partial)$	$O(qs_{\mathrm{o}} + \partial)$
θ	$O(qs_{\mathrm{o}} \cdot ls_{\mathrm{o}} \cdot S_m)$	$O(qs_{\mathrm{o}} \cdot ls_{\mathrm{o}} \cdot S_m)$

where vdg_{o}, var_{o}, id_{o}, qs_{o}, ls_{o} are maxima taken over all the axioms of $\mathcal{P}$ [11] of vdg, var, id, qs and ls respectively and ld_1, ls_1 are maxima of ld, ls taken over contractions $A \rightarrow A \wedge A$ of $\mathcal{P}$.

[10] Since functional interpretation treats $\vee$ as an existential quantifier we count it as quantifier.

[11] In fact it is sufficient to consider only the axioms of the transformed proof $\mathcal{P}^{\mathrm{tr}}$, see Definition 3.8. Valid also for the subsequent definitions, including the classical case.

For **classical** proofs $\mathcal{P}$ there exists $k \in \mathbb{N}$ constant (independent of $\mathcal{P}$) such that (below 'FI' means 'functional interpretation'):

	usual FI	monotone FI
mdg	$vdg_{\mathrm{o}} + O(fid_{\mathrm{o}})$	$vdg_{\mathrm{o}} + O(fid_{\mathrm{o}})$
mar	$var_{\mathrm{o}} + O(qs_{\mathrm{o}} \cdot fid_{\mathrm{o}})$	$var_{\mathrm{o}} + O(qs_{\mathrm{o}} \cdot fid_{\mathrm{o}})$
d	$O(ls_{\mathrm{o}} \cdot \partial)$	$O(qs_{\mathrm{o}} \cdot \partial)$
S	$O(S_c)\,,\, O(ls_{\mathrm{o}} \cdot qs_{\mathrm{o}}^{k \cdot \partial})$	$O(S_m)\,,\, O(qs_{\mathrm{o}}^{k \cdot \partial})$
∂_v	$O(ls_{\mathrm{o}} + \partial)$	$O(qs_{\mathrm{o}} + \partial)$
θ	$O(qs_{\mathrm{o}} \cdot ls_{\mathrm{o}} \cdot S_m)$	$O(qs_{\mathrm{o}} \cdot ls_{\mathrm{o}} \cdot S_m)$

where vdg_{o}, var_{o}, qs_{o}, ls_{o} are maxima taken over all the axioms of $\mathcal{P}$ of vdg, var, qs and ls respectively and fid_{o} is the maximum of fid over all the formulas of $\mathcal{P}$.

Since they are not produced by functional interpretation, we normally do not count the terms t_1, t_2 which appear in prime formulas $t_1 = t_2$ of contractions $A \to A \wedge A$ and the quantifier axioms terms as part of the realizing terms. We rather consider them as "black boxes" and use their type and free variables information only (see Definition 3.10). From a programming perspective, they may be considered as subprograms residing in libraries and made accessible to the extracted program via references. The bounds for the usual functional interpretation actually hold also if we count in the terms mentioned above provided that instead of ld, ls one uses wd, respectively ws, where

- wd is the whole depth of A, assuming a tree representation of A where tree representations of the terms occurring in A are linked from the corresponding leaves of the usual tree representation of A;
- ws is the whole size of A, i.e., the number of all logical constants of A plus the number of all occurrences of variables and constants in A.

For mdg and mar also the maximal degree, respectively arity of constants occurring in contraction and quantifier axioms terms must be counted in. For more details see Remark 3.28.

1.2 Notational conventions

The symbols $:\equiv$ and $\equiv$ belong to the meta-level and mean *equal by definition to* and *is identical to* respectively. The symbol $=$ is used by abuse for equality in both meta-level and formal systems.

For a set M we let $M^{\leq\omega} :\equiv \cup_{n\leq\omega} M^n$. The symbol $\mathbb{N}$ denotes the set of *natural numbers*. For a function $f : M' \mapsto \mathbb{N}$ and $M \subseteq M'$, M finite, we let

$$f(M) :\equiv max\{f(m) \mid m \in M\}.$$

An enumeration $\mathrm{S}_1, \ldots, \mathrm{S}_n$ denotes an ordered tuple abbreviated $\underline{\mathrm{S}}$; we denote by $\{\underline{\mathrm{S}}\}$ the set corresponding to $\underline{\mathrm{S}}$, by $|\underline{\mathrm{S}}|$ the length of $\underline{\mathrm{S}}$ and by $\underline{\mathrm{S}'}, \underline{\mathrm{S}''}$ the concatenation of $\underline{\mathrm{S}'}$ and $\underline{\mathrm{S}''}$. If $\{\underline{\mathrm{S}}\} \subseteq M'$ we abbreviate by $f(\underline{\mathrm{S}}) :\equiv f(\{\underline{\mathrm{S}}\})$.

Let $k_0 \in \mathbb{N}$ be a sufficiently large constant ($k_0 \equiv 10$ suffices for our purposes). For a labeled tree $\triangle$ we denote by $\partial(\triangle)$ the depth of $\triangle$ plus k_0, by $\partial_L(\triangle)$ the L depth of $\triangle$, namely the maximal number of L labels on a path from root to leaves plus k_0, by $Lv(\triangle)$ the set of labels of leaves of $\triangle$ and by $Vt(\triangle)$ the set of labels of all vertices of $\triangle$.

A *(formal) proof* in some logical system is a tree whose vertices are labeled with formulas, such that the leaves are labeled with axioms and assumptions and any parent vertex is labeled with the result of the application of an instance of some rule to the labels of its sons. The edges which connect the parent vertex with its sons are labeled with the name of the corresponding rule. We denote by $L(\cdot)$ the *labeling function* on vertices and edges. We call a proof *complete* if all its leaves are labeled with axioms only. Notice that an *incomplete* proof is complete in the system extended with its assumptions as axioms. We will denote proofs by $\vdash$ or ——, possibly with bounds on the depth attached, such as $\vdash_n$ for a proof of depth at most n, $n \in \mathbb{N}$.

2 The weak base system $\mathtt{EIL}^\omega$

In the following we introduce the system $\mathtt{EIL}^\omega$ [12] which forms in a sense a weak base system containing exactly the tools needed to carry out the functional interpretation. It extends intuitionistic logic in finite types with appropriate combinators[13], a cases operator D and some very basic arithmetic needed to define characteristic functionals for quantifier-free formulas.

[12] A short for '(weakly extensional) extended intuitionistic logic in all finite types'.

[13] These allow the definition of λ-terms, see Definition 2.12.

We also include C. Spector's quantifier-free rule of extensionality $\mathtt{ER}_0$. This allows an as extensional as possible treatment of higher type equality in the context of functional interpretation[14].

We first carry out a full quantitative analysis for the functional interpretation of an extension $\mathtt{EIL}^\omega_+ + \mathtt{AC} + \mathtt{IP}_\forall + \mathtt{MK}$ [15] of $\mathtt{EIL}^\omega$ into the quantifier-free fragment of $\mathtt{EIL}^\omega$. Due to the modularity of functional interpretation this analysis immediately relativizes to further extensions of $\mathtt{EIL}^\omega$ with certain axioms like, e.g., induction. Suppose that we consider an additional (closed) axiom A. Let us add to $\mathtt{EIL}^\omega$ new constants $\underline{c}$ of appropriate types and the axiom[16] $\forall \underline{y} A_\mathtt{D}(\underline{c}, \underline{y})$ expressing that $\underline{c}$ satisfies the functional interpretation of A. The quantitative analysis for the functional interpretation of $\mathtt{EIL}^\omega_+ + \mathtt{AC} + \mathtt{IP}_\forall + \mathtt{MK}$ immediately relativizes to this extension. Functional interpretation now provides realizing terms $\underline{t}[\underline{c}]$ built up out of the $\mathtt{EIL}^\omega$-material and $\underline{c}$. The complexity analysis for the extended theory is then completed by determining actual terms $\underline{s}$ which satisfy the functional interpretation $\exists \underline{x} \forall \underline{y} A_\mathtt{D}(\underline{x}, \underline{y})$ of A and the complexity of the verifying proof $\vdash \forall \underline{y} A_\mathtt{D}(\underline{s}, \underline{y})$.

There are two possible ways of handling λ-abstraction in a system like $\mathtt{EIL}^\omega$. We could treat λ-abstraction either as a primitive concept or as defined by combinators. The treatment via combinators provides a finer complexity analysis and reflects more faithfully the actual functional interpretation of a Hilbert-style axiomatization[17] of intuitionistic logic which we have – following Gödel's original formulation – used for $\mathtt{EIL}^\omega$.

The combinators and projectors we use are more flexible than the usual Σ and Π first introduced by Schönfinkel in [47]. Our Σ provide in particular

[14] Most applications of functional interpretation have been based on such an extensional variant. For sentences containing only variables of type 0 or 1 the use of full extensionality is admissible since the elimination-of-extensionality procedure from [42] is applicable.

[15] Here $\mathtt{AC}$ is the Axiom of Choice, $\mathtt{IP}_\forall$ is Independence of Premises for universal premises and $\mathtt{MK}$ is a variant of Markov's principle, see Section 3.1. For $\mathtt{EIL}^\omega_+$ see Definition 3.10.

[16] Here $\exists \underline{x} \forall \underline{y} A_\mathtt{D}(\underline{x}, \underline{y})$ is the functional interpretation of A, see also Section 3.

[17] In a natural deduction context, it might be more natural to treat λ-abstraction as a primitive concept. Natural deduction formulations of functional interpretation are provided by Diller-Nahm [13] (see also [46, 51]) and Joergensen [25]. In the former all cases definitions for realizing terms of contractions are postponed to the end by collecting all candidates and making a single final global choice. In the latter choices are local and one has to apply a so-called 'contraction lemma' for each of them, i.e., whenever more than one copy of an assumption gets cancelled. In any case, the analysis carried out in the present paper can immediately be adapted to a system with λ-abstraction included as primitive construct, see Remark 3.30.

extensions of Schönfinkel Σ to tuples (see Definition 2.4) and our Π are extensions of Schönfinkel Π to tuples. This is natural since we use tuples of variables throughout our formulation of functional interpretation. The design of our Σ and Π is made according to the actual constructs required by functional interpretation while keeping the benefits of the usual Σ and Π. The latter allow to avoid any notion of bound variables in terms and are the most convenient in connection with logical relations[18]. Our Σ and Π are in fact definable in terms of usual Σ and Π, though at the expense of a rather artificial increase in the length of the verifying proof. The upper bound on the size of the extracted terms would nevertheless still hold with such a definition, see Remark 3.30.

2.1 The type structure FT

The set FT of all *finite types* is inductively generated by the rules

(i) $o \in \mathtt{FT}$

(ii) If $\sigma, \tau \in \mathtt{FT}$ then $(\sigma\tau) \in \mathtt{FT}$.

Intuitively type o represents the set of natural numbers and $(\sigma\tau)$ represents the set of functions which map objects of type σ to objects of type τ. There are many alternative notations in the literature for $(\sigma\tau)$, such as $\tau(\sigma)$, $(\sigma)\tau$, $(\sigma \to \tau)$. We make the convention that concatenation of types is right associative and consequently omit unnecessary parenthesis, writing $\delta\sigma\tau$ instead of $(\delta(\sigma\tau))$. It can immediately be verified by induction over FT that each $\sigma \in \mathtt{FT}$ has the form $\sigma_1 \dots \sigma_n o$ with $n \geq 0$. We abbreviate by:

- $\underline{\sigma}$ the ordered tuple of types $\sigma_1, \dots, \sigma_n$
- $\underline{\sigma}\tau$ the type $\sigma_1 \dots \sigma_n \tau$.

If p is a permutation of $\{1, \dots, n\}$, $\underline{\sigma}^p$ abbreviates the tuple $\sigma_{p_1}, \dots, \sigma_{p_n}$.

Definition 2.1 For a type we define:

- the *arity* by $ar(o) :\equiv 0$, $ar(\sigma\tau) :\equiv ar(\tau) + 1$;
- the *degree* by $dg(o) :\equiv 0$, $dg(\sigma\tau) :\equiv max\{dg(\sigma) + 1\,,\, dg(\tau)\}$.

[18] One example of a logical relation is Howard's majorizability which plays a key role in most applications of functional interpretation [3, 26, 27, 28, 30, 31, 33, 35, 38].

Let

$$\begin{aligned} dg(\underline{\sigma}) &:\equiv max\{dg(\sigma_1)\,,\,\ldots\,,\,dg(\sigma_n)\} \\ ar(\underline{\sigma}) &:\equiv max\{ar(\sigma_1)\,,\,\ldots\,,\,ar(\sigma_n)\}\;. \end{aligned}$$

The following hold:

$$\begin{aligned} dg(\underline{\sigma}\tau) &= max\{dg(\underline{\sigma})+1\,,\,dg(\tau)\} \\ ar(\underline{\sigma}\tau) &= ar(\tau)+|\underline{\sigma}| \end{aligned}$$

2.2 Intuitionistic Equality Logic over FT ($\mathtt{IEL}^\omega$)

Our formalization of $\mathtt{IEL}^\omega$ below is a slight modification of the axiomatic calculus for multisorted intuitionistic predicate logic used by Gödel in his original paper on functional interpretation [21]. The only differences are:

1. The *syllogism* and *expansion* are formulated as axioms instead of rules. Gödel's formulation with rules was designed to ease the formulation of the soundness proof for the functional interpretation. Nevertheless for the quantitative analysis it is more convenient to use the axiom versions of

   (a) the expansion rule
   $$\frac{A\to B}{C\vee A\to C\vee B}\,,$$
   since the formula C may introduce realizing terms of arbitrary complexities; also the formula complexity of the conclusion is higher than the one of the premise;

   (b) the syllogism rule
   $$\frac{A\to B\,,\,B\to C}{A\to C}\,,$$
   which would force us to consider the sum of quantitative measures of both premises when computing upper bounds for quantitative measures of the conclusion. We can immediately notice that the mere Modus Ponens
   $$\frac{A\,,\,A\to B}{B}$$
   avoids such a situation, since the formula complexity of the premise $A\to B$ upper bounds that of the conclusion B.

2. The quantifier rules and axioms are formulated with tuples of variables since we use tuples throughout the functional interpretation.

The language of $\mathtt{IEL}^\omega[\mathrm{C}]$ contains, aside from the constants C, the following:

- denumerably many *variables* which we denote by letters x, y, z, u, v, w, possibly capitalized or adorned with subscripts; $\underline{x} :\equiv x_1, \ldots, x_n$ denotes a tuple of variables; in the same context we use x as metavariable for an individual element of $\underline{x}$; each of the variables is associated a unique sort (mostly called type) which is an element of $\mathtt{FT}$, such that there exist denumerably many variables for each sort; we possibly indicate the type of a variable by carrying it as a superscript, like x^σ and then $\underline{x^\sigma} :\equiv x_1^{\sigma_1}, \ldots, x_n^{\sigma_n}$.
- a binary predicate constant $=_o$ for equality between objects of type o;
- logical constants $\bot,\ \wedge,\ \vee,\ \rightarrow,\ \forall x$ and $\exists x$ (for each variable x).

Each of the constants in C is sorted as well, with the type possibly indicated as superscript. We often do not indicate C and write $\mathtt{IEL}^\omega$ when the set of constants is either clear from the context or not relevant. We use l as metavariable for both variables and constants.

The terms of $\mathtt{IEL}^\omega$ are sorted, with their types possibly indicated in superscripts and are inductively generated from variables and constants according to the rule that if $t^{\sigma\tau}$ and s^σ are terms then $(ts)^\tau$ is a term.

Terms are denoted by letters s, t, r, possibly adorned with subscripts; tuples of terms are denoted like $\underline{t} :\equiv t_1, \ldots, t_n$; in the same context we use t as metavariable for an individual element of $\underline{t}$.

We denote by $\mathrm{V}(t)$ the set of variables occurring in t and write $t[\underline{x}]$ to indicate that $\{\underline{x}\} \subseteq \mathrm{V}(t)$. If $\mathrm{V}(t) = \emptyset$ we say that t is a *closed* term.

We make the convention that concatenation of terms is left associative and consequently omit unnecessary parenthesis, writing rst instead of $((rs)t)$.

When writing down an expression it is always assumed that the terms are well-formed, i.e. the types are fitting.

For t^σ we denote by $\mathit{typ}(t) :\equiv \sigma$ and by

- $ar(t) :\equiv ar(\sigma)$ the *arity* of t;
- $dg(t) :\equiv dg(\sigma)$ the *degree* of t.

For a term we define

- the *depth* by $d(l) :\equiv 0$, $d(ts) :\equiv max\{d(t)\,,\, d(s)\} + 1$
- the *size* by $S(l) :\equiv 1$, $S(ts) :\equiv S(t) + S(s)$.

The *subterm* relation is defined as the reflexive transitive closure of $\{(s, ts), (t, ts)\}$. We denote by $s \leq t$ the fact that s is a subterm of t. Obviously $\leq$ is a partial order relation. Let

$$\begin{array}{lcl} mdg(t) & :\equiv & max\{dg(s) \mid s \leq t\} \\ mar(t) & :\equiv & max\{ar(s) \mid s \leq t\}. \end{array}$$

We notice that:

- $dg(t) \geq dg(ts)$, hence $mdg(r) = max_{l \leq r}\, dg(l)$
- $ar(t) \geq ar(ts)$, hence $mar(r) = max_{l \leq r}\, ar(l)$

We abbreviate by:

$$\begin{array}{lcl} t(\underline{s}) & :\equiv & t\, s_1 \ldots s_m \\ \underline{t}(\underline{s}) & :\equiv & t_1(\underline{s}), \ldots, t_n(\underline{s}) \end{array}$$

The formulas of $\mathtt{IEL}^\omega$ are inductively generated from *prime formulas* $s^o =_o t^o$ and $\bot$ according to the rule that if A and B are formulas then $(A \wedge B)$, $(A \vee B)$, $(A \to B)$, $(\forall x A)$ and $(\exists x A)$ are formulas.

Equality between terms s, t of type $\sigma = \sigma_1 \ldots \sigma_n o$ $(1 \leq n)$ is an abbreviation for $\forall x_1^{\sigma 1} \ldots x_n^{\sigma_n}(s\, x_1 \ldots x_n =_o t\, x_1 \ldots x_n)$, where the variables $x_1, \ldots, x_n$ do not occur in s, t. We abbreviate by $\underline{s} = \underline{t} :\equiv (s_1 = t_1), \ldots, (s_n = t_n)$ (hence a tuple of formulas).

$\forall \underline{x}$, $\exists \underline{x}$ abbreviate $\forall x_1 \ldots \forall x_n$ and $\exists x_1 \ldots \exists x_n$ respectively.

We abbreviate by $A \leftrightarrow B :\equiv ((A \to B) \wedge (B \to A))$, $\neg A :\equiv (A \to \bot)$ and $s \neq t :\equiv \neg(s = t)$. In order to avoid unnecessary parenthesis we make the convention that $\forall x$, $\exists x$, $\neg$, $\wedge$, $\vee$, $\to$, $\leftrightarrow$ is the decreasing order of precedence and that $\to$ is right associative.

We denote formulas by letters A, B, C, possibly adorned with subscripts or superscripts. We call a formula *quantifier-free* if it does not contain $\forall, \exists, \vee$. The subscript 0 always indicates a quantifier-free formula, such as A_0, B_0, C_0.

We denote by $\mathrm{V_f}(A)$, $\mathrm{V_b}(A)$, $\mathrm{V}(A)$ the set of free, bounded, respectively all variables occurring in A and write $A(\underline{x})$ to indicate that $\{\underline{x}\} \subseteq \mathrm{V_f}(A)$. We

denote by $\mathrm{C}(A)$ the set of constants occurring in A and by $vdg(A) :\equiv dg(\mathrm{V}(A))$, $var(A) :\equiv ar(\mathrm{V}(A))$, $cdg(A) :\equiv dg(\mathrm{C}(A))$, $car(A) :\equiv ar(\mathrm{C}(A))$.

For $\mathrm{S} \subseteq \{\forall, \exists, \wedge, \vee, \rightarrow\}$ we define $d_{\mathrm{S}}(\cdot)$, the S-*depth* of a formula by

- $d_{\mathrm{S}}(s =_o t) :\equiv d_{\mathrm{S}}(\bot) :\equiv k_0$ (see Section 1.2 for k_0)
- For $Q \in \{\forall, \exists\}$, $d_{\mathrm{S}}(Qx\, A) :\equiv \begin{cases} d_{\mathrm{S}}(A) + 1, \text{ if } Q \in \mathrm{S} \\ d_{\mathrm{S}}(A) \quad\quad , \text{ if } Q \notin \mathrm{S} \end{cases}$
- For $\square \in \{\wedge, \vee, \rightarrow\}$, $d_{\mathrm{S}}(A \square B) :\equiv \begin{cases} max\{d_{\mathrm{S}}(A)\,,\, d_{\mathrm{S}}(B)\} + 1, \text{ if } \square \in \mathrm{S} \\ max\{d_{\mathrm{S}}(A)\,,\, d_{\mathrm{S}}(B)\} \quad\quad , \text{ if } \square \notin \mathrm{S} \end{cases}$

For a formula A we define the following:

- the *logical constants depth* by $ld(A) :\equiv d_{\forall,\exists,\wedge,\vee,\rightarrow}(A)$;
- the *whole depth* by $wd(A) :\equiv d'_{\forall,\exists,\wedge,\vee,\rightarrow}(A)$; here d' differs from d just in $d'_{\mathrm{S}}(s =_o t) :\equiv k_0 + max\{d(s)\,,\, d(t)\}$
- the *implication depth* by $id(A) :\equiv d^{\mathrm{o}}_{\rightarrow}(A)$ and the *forall depth* by $fd(A) :\equiv d^{\mathrm{o}}_{\forall}(A)$; here d^{o} differs from d just in $d^{\mathrm{o}}_{\mathrm{S}}(A_0) :\equiv k_0$;
- the *forall/implication depth* by $fid(A) :\equiv max\{fd(A)\,,\, id(A)\}$;
- the *quantifier size*, denoted $qs(A)$, is the number of quantifiers (including $\vee$) occurring in A, when A is a closed formula and the quantifier size of its universal closure in the general case;
- the *logical constants size*, denoted $ls(A)$, is obtained by adding to $qs(A)$ the number of $\wedge, \rightarrow, \bot, =$ occurring in A;
- the *whole size*, denoted $ws(A)$, is obtained by adding to $ls(A)$ the number of all occurrences of variables and constants in A.

We present below the rules and axioms of $\mathtt{IEL}^{\omega}$.

Logical rules

MP :	$A\,,\, A \to B \vdash B$	(modus ponens)
EXP :	$A \wedge B \to C \vdash A \to (B \to C)$	(exportation)
IMP :	$A \to (B \to C) \vdash A \wedge B \to C$	(importation)
QR$\forall$:	$B \to A \vdash B \to \forall \underline{z} A$	(quantifier rules)
QR$\exists$:	$A \to B \vdash \exists \underline{z} A \to B$	

We denote by $\mathtt{QR} :\equiv \mathtt{QR}\forall + \mathtt{QR}\exists$. At QR, z is not free in B.

Logical axioms

CT$\vee$:	$A \vee A \to A$	(contraction)
CT$\wedge$:	$A \to A \wedge A$	
WK$\vee$:	$A \to A \vee B$	(weakening)
WK$\wedge$:	$A \wedge B \to A$	
PM$\vee$:	$A \vee B \to B \vee A$	(permutation)
PM$\wedge$:	$A \wedge B \to B \wedge A$	
SYL :	$(A \to B) \wedge (B \to C) \to (A \to C)$	(syllogism)
EPN :	$(A \to B) \to (C \vee A \to C \vee B)$	(expansion)
EFQ :	$\bot \to A$	(ex falso quodlibet)
QA$\forall$:	$\forall \underline{z} A(\underline{z}) \to A(\underline{s})$	(quantifier axioms)
QA$\exists$:	$A(\underline{s}) \to \exists \underline{z} A(\underline{z})$	

We denote by $\mathtt{QA} :\equiv \mathtt{QA}\forall + \mathtt{QA}\exists$. At QA, s is free for z in A and the substitution is simultaneous.

For instances $B(\underline{s})$ of QA which involve the constants $\underline{s}$, we define the

- *term depth* of $B(\underline{s})$ by $td(B) :\equiv d(\underline{s})$;
- *term size* of $B(\underline{s})$ by $ts(B) :\equiv \Sigma_{s \in \underline{s}} S(s)$.

Equality axioms

REF :	$x =_o x$	(reflexivity)
SYM :	$x =_o y \to y =_o x$	(symmetry)
TRZ :	$x =_o y \wedge y =_o z \to x =_o z$	(transitivity)

Remark 2.2 There exists $k \in \mathbb{N}$ constant such that for all σ :

Higher-order equality

REF$[\sigma]$:	$\mathtt{IEL}^\omega \vdash_k x =_\sigma x$	(reflexivity)
SYM$[\sigma]$:	$\mathtt{IEL}^\omega \vdash_k x =_\sigma y \to y =_\sigma x$	(symmetry)
TRZ$[\sigma]$:	$\mathtt{IEL}^\omega \vdash_k x =_\sigma y \wedge y =_\sigma z \to x =_\sigma z$	(transitivity)

Given a set of rules (axioms are comprised as rules with empty premise) Rl whose formulas contain the constants C, we denote by $\mathtt{IEL}^\omega$[Rl] the system $\mathtt{IEL}^\omega$[C] extended with the rules in Rl. We sometimes abbreviate $\mathtt{IEL}^\omega$[Rl] with a different denotation (like $\mathtt{EIL}^\omega$ below) and then $(\mathtt{IEL}^\omega[\mathtt{Rl}])[\mathtt{Rl}'] :\equiv \mathtt{IEL}^\omega[\mathtt{Rl} \cup \mathtt{Rl}']$.

2.3 The system $\mathtt{EIL}^\omega$

Multisorted weakly extensional extended intuitionistic equality logic over FT, which we denote by $\mathtt{EIL}^\omega$, is obtained by extending $\mathtt{IEL}^\omega$ with exactly the elements which are strictly necessary to carry out functional interpretation even for $\mathtt{IEL}^\omega$.

The language of $\mathtt{EIL}^\omega$ contains the following constants:

- the *zero* constant $0 \equiv O_o$ of type o and for each type $\rho \equiv \underline{\sigma} o$, the *zero* constant O_ρ defined by the axiom

  AxO : $\quad O_\rho(\underline{z}^{\underline{\sigma}}) = 0$

  (hence for any type there exists at least one constant)

- the *successor* constant S of type oo defined by the axiom

  AxS : $\quad \left\{ \begin{array}{c} S\,x \neq 0 \\ S\,x = S\,y \to x = y \end{array} \right.$

- the *boolean* constants ν, I, E all of type *ooo* defined by the axioms

  $\mathtt{Ax}\nu:\quad (x = 0 \wedge y = 0) \leftrightarrow \nu\, x\, y = 0$

  $\mathtt{Ax}I:\quad (x = 0 \to y = 0) \leftrightarrow I\, x\, y = 0$

  $\mathtt{Ax}E:\quad x = y \leftrightarrow E\, x\, y = 0$

- for each

  - $n, m \in \mathbb{N}$, $\underline{n} :\equiv n_0, n_1, \ldots, n_m \in \mathbb{N}$, $\overline{n} :\equiv n^1, \ldots, n^m \in \mathbb{N}$ with $n_0, n_1, \ldots, n_m \leq n$ and $n^1, \ldots, n^m \leq n$,
  - permutations $\underline{p} :\equiv p_0, p_1, \ldots, p_m$ and $\overline{p} :\equiv p^1, \ldots, p^m$ of $\{1, \ldots, n\}$
  - types τ, $\underline{\sigma} \equiv \sigma_1, \ldots, \sigma_n$, $\underline{\delta} \equiv \delta_1, \ldots, \delta_m$

  the *combinator* constant $\Sigma^{\underline{\sigma},\underline{\delta},\tau,m}_{\underline{p},\overline{p},\underline{n},\overline{n}}$ of type

  - $(\underline{\sigma_0}\, \delta_1\, \underline{\sigma_1}\, \ldots\, \delta_m\, \underline{\sigma_m}\, \tau)\, (\underline{\sigma}^1\, \delta_1) \ldots (\underline{\sigma}^m\, \delta_m)\, \underline{\sigma}\, \tau$,

  where

  - $\underline{\sigma_j} :\equiv \sigma_{(p_j)_1}, \ldots, \sigma_{(p_j)_{n_j}}$ with $j \in \overline{0, m}$ and
  - $\underline{\sigma}^j :\equiv \sigma_{(p^j)_1}, \ldots, \sigma_{(p^j)_{n^j}}$ with $j \in \overline{1, m}$

  defined by the axiom

  $\mathtt{Ax}\Sigma:\quad \Sigma^{\underline{\sigma},\underline{\delta},\tau,m}_{\underline{p},\overline{p},\underline{n},\overline{n}}(x, \underline{y}, \underline{z}) = x(\underline{z_0}, y_1(\underline{z}^1), \underline{z_1}, \ldots, y_m(\underline{z}^m), \underline{z_m})$

- for each $n \in \mathbb{N}$, permutation p of $\{1, \ldots, n\}$ and types $\tau, \underline{\sigma} \equiv \sigma_1, \ldots, \sigma_n$, the *permutation* constant $P^{\underline{\sigma},\tau}_{n,p}$ of type $(\underline{\sigma}\tau)\underline{\sigma}^p\tau$ defined by the axiom

  $\mathtt{Ax}P:\quad P^{\underline{\sigma},\tau}_{n,p}(x, \underline{z'}) = x(\underline{z})$

- for each $n, i \in \mathbb{N}$, $i \leq n$ and types $\underline{\sigma} \equiv \sigma_1, \ldots, \sigma_n$, the *projector* constant $\Pi^{\underline{\sigma}}_i$ of type $\underline{\sigma}\sigma_i$ defined by the axiom

  $\mathtt{Ax}\Pi:\quad \Pi^{\underline{\sigma}}_i(\underline{z}) = z_i$

- for each $n, i \in \mathbb{N}$, $i \leq n$ and types $\underline{\sigma} \equiv \sigma_1, \ldots, \sigma_n$, the *decision* constant $D^{\underline{\sigma}}_i$ of type $o\underline{\sigma}\,\underline{\sigma}\sigma_i$ defined by the axiom

  $\mathtt{Ax}D:\quad \begin{cases} x = 0 \to D^{\underline{\sigma}}_i(x, \underline{z}, \underline{z'}) = z_i \\ x \neq 0 \to D^{\underline{\sigma}}_i(x, \underline{z}, \underline{z'}) = {z'}_i \end{cases}$ Here $|\underline{z}| = |\underline{z'}|$.

For simplicity we abbreviate by $1 :\equiv S0$. The system $\mathtt{EIL}^\omega$ is finally obtained by adding the quantifier-free tertium non datur axiom

$\mathtt{TND}_0 :\qquad x = 0 \vee \neg(x = 0)$

and the quantifier-free extensionality rule

$$\mathtt{ER}_0 :\qquad \frac{A_0 \to s_1 = t_1\ ,\ \ldots\ ,\ A_0 \to s_n = t_n}{A_0 \to B_0(\underline{s}) \to B_0(\underline{t})}\ .$$

The formal proofs in the sequel will be in $\mathtt{EIL}^\omega$ if not otherwise indicated.

Remark 2.3 The constants P and Π are definable in terms of Σ and also $O_{\underline{\sigma}o} = \Pi_1^{o,\underline{\sigma}}0$. We nevertheless chose to define them separately since they play a particular role.

Definition 2.4 As particular cases of Σ we distinguish the *tuple-Schönfinkel combinators* $\Sigma^{\underline{\sigma},(\delta),\tau,1}_{(\mathtt{1_n},\mathtt{1_n}),(\mathtt{1_n}),(n,0),(n)}$ with defining axioms of shape

$$\Sigma^{\underline{\sigma},(\delta),\tau,1}_{(\mathtt{1_n},\mathtt{1_n}),(\mathtt{1_n}),(n,0),(n)}(x,y,\underline{z}) = x(\underline{z}, y(\underline{z}))\ .$$

These are generalizations of the usual[19] Schönfinkel combinators Σ to tuples and will be used in the λ-abstraction Definition 2.12. The usual *Schönfinkel combinators* Σ are in fact particular cases of our Σ of shape $\Sigma^{\underline{\sigma},(\delta),\tau,1}_{(\mathtt{1_1},\mathtt{1_1}),(\mathtt{1_1}),(1,0),(1)}$ with defining axioms $\Sigma^{\underline{\sigma},(\delta),\tau,1}_{(\mathtt{1_1},\mathtt{1_1}),(\mathtt{1_1}),(1,0),(1)}(x,y,z) = x(z, y(z))$. Also the usual *Schönfinkel projectors* Π are particular cases of our Π of shape $\Pi_1^{(\sigma_1,\sigma_2)}$ with defining axioms $\Pi_1^{(\sigma_1,\sigma_2)}(z_1,z_2) = z_1$.

Remark 2.5 The quantifier-free tertium-non-datur $\mathtt{TND}_0$ becomes derivable in the presence of induction for propositional formulas. Moreover, in the presence of a modest amount of arithmetic, the constants D, I, ν, E are definable and their axioms derivable. Therefore these axioms are in fact redundant in any concrete application of functional interpretations, e.g., to $\mathtt{HA}^\omega$ and fragments thereof. Examples of the latter are systems of bounded arithmetic like $\mathtt{IPV}^\omega$ of [10] and the poly-time arithmetic LHA of [48] [20] .

[19] For the original definition of Schönfinkel Σ and Π see [47]. See also the last paragraph before Section 2.1.

[20] Even though LHA was designed in a modified realizability context, the outline of a similar functional-interpretation-generated system should be straightforward.

Remark 2.6 The extensionality axiom

$$\mathtt{EA}[\underline{\sigma}]: \qquad \underline{x}^{\underline{\sigma}} = \underline{y}^{\underline{\sigma}} \to f^{\underline{\sigma}o}\,\underline{x} =_o f^{\underline{\sigma}o}\,\underline{y} \qquad (\text{ let } \mathtt{EA} :\equiv \cup_{\underline{\sigma}} \mathtt{EA}[\underline{\sigma}])$$

is derivable in $\mathtt{EIL}^\omega$ for $\underline{\sigma} \equiv o, \ldots, o$ hence $\mathtt{EIL}^\omega$ contains *all* equality axioms for type o. This no longer holds in general when $\underline{\sigma}$ contains higher types (follows from [53](3.5.10) and [24]). On the other hand, $\mathtt{ER}_0$ is derivable from $\mathtt{EA}$ in $\mathtt{EIL}^\omega \setminus \mathtt{ER}_0$, hence the rule is strictly weaker than the axiom, but only at higher types.

Remark 2.7 $\vdash \bot \leftrightarrow 1 = 0$ and $\vdash x \neq 0 \leftrightarrow Ix1 = 0$.

Remark 2.8 There exists $k \in \mathbb{N}$ constant such that for all $\underline{s}$, $\underline{t}$, r, r_1, r_2, B_0, the following hold:

$$\underline{s} = \underline{t} \;\vdash_k\; B_0(\underline{s}) \to B_0(\underline{t}) \tag{1}$$

$$\underline{s} = \underline{t} \;\vdash_k\; r[\underline{s}] = r[\underline{t}]$$

$$r_1 = r_2, \underline{s} = \underline{t} \;\vdash_k\; r_1(\underline{s}) = r_2(\underline{t}). \tag{2}$$

Proposition 2.9

$$\begin{aligned}
dg(\Sigma^{\underline{\sigma},\underline{\delta},\tau,m}_{\underline{p},\overline{p},\underline{n},\overline{n}}) &= max\{dg(\underline{\sigma},\underline{\delta}) + 2\,,\, dg(\tau) + 1\} \\
dg(\Pi^{\underline{\sigma}}_i) &= dg(\underline{\sigma}) + 1 \\
dg(D^{\underline{\sigma}}_i) &= dg(\underline{\sigma}) + 1 \\
dg(P^{\underline{\sigma},\tau}_{n,p}) &= max\{dg(\underline{\sigma}) + 2\,,\, dg(\tau) + 1\}
\end{aligned}$$

$$\begin{aligned}
ar(\Sigma^{\underline{\sigma},\underline{\delta},\tau,m}_{\underline{p},\overline{p},\underline{n},\overline{n}}) &= ar(\tau) + |\underline{\sigma}| + |\underline{\delta}| + 1 \\
ar(\Pi^{\underline{\sigma}}_i) &= ar(\sigma_i) + |\underline{\sigma}| \\
ar(D^{\underline{\sigma}}_i) &= ar(\sigma_i) + 2|\underline{\sigma}| + 1 \\
ar(P^{\underline{\sigma},\tau}_{n,p}) &= ar(\tau) + |\underline{\sigma}| + 1
\end{aligned}$$

□

In the proposition below we show how and at which cost in proof depth the quantifier-free formulas can be viewed as prime formulas.

Proposition 2.10 (Association of terms to quantifier-free formulas) There exists $k \in \mathbb{N}$ constant and an association of terms to quantifier-free formulas $A_0 \mapsto t_{A_0}$ such that for all A_0,

$$\vdash_{k \cdot ld(A_0)} \quad A_0(\underline{a}) \leftrightarrow t_{A_0}[\underline{a}] = 0\ . \tag{3}$$

Proof: The proof is by induction on the structure of A_0, making use of the boolean constants axioms. For prime formulas just take $t_\perp :\equiv 1$, $t_{t_1=t_2} :\equiv E t_1 t_2$ then recursively define $t_{B_0 \wedge C_0} :\equiv \nu t_{B_0} t_{C_0}$, $t_{B_0 \to C_0} :\equiv I t_{B_0} t_{C_0}$. $\square$

Corollary 2.11 (TND and Stability for quantifier-free formulas) There exists $k \in \mathbb{N}$ constant such that for all A_0,

$$\vdash_{k \cdot ld(A_0)} \quad A_0(\underline{a}) \ \vee \ \neg A_0(\underline{a}) \tag{4}$$

$$\vdash_{k \cdot ld(A_0)} \quad \neg\neg A_0(\underline{a}) \ \to \ A_0(\underline{a}) \ . \tag{5}$$

Proof: The principle

$\mathtt{STAB_0}$: $\quad \neg\neg\, x = 0 \ \to \ x = 0$

follows immediately with constant-depth proof from $\mathtt{TND_0}$. Both (4) and (5) follow immediately from $\mathtt{TND_0}$ and $\mathtt{STAB_0}$ respectively by (3) and (1). $\square$

Definition 2.12 (λ-abstraction) To every term t^τ one associates a term $(\lambda \underline{x}^\sigma . t)^{\sigma\tau}$, with $\mathrm{V}(\lambda \underline{x}.\, t) = \mathrm{V}(t) - \{\underline{x}\}$, recursively defined as follows:

$$\begin{array}{rcl} \lambda \underline{x}.\, x_i & :\equiv & \Pi_i^\sigma \\ \lambda \underline{x}.\, t & :\equiv & \Pi_1^{(\tau,\sigma)} t, \ \text{if } \{\underline{x}\} \cap \mathrm{V}(t) = \emptyset \\ \lambda \underline{x}.\, (t^{\delta\tau} s^\delta) & :\equiv & \Sigma^{\sigma,(\delta),\tau,1}_{(1_{\mathtt{n}},1_{\mathtt{n}}),(1_{\mathtt{n}}),(n,0),(n)} (\lambda \underline{x}.\, t)(\lambda \underline{x}.\, s), \ \text{if } \{\underline{x}\} \cap \mathrm{V}(ts) \neq \emptyset \end{array} \tag{6}$$

Proposition 2.13 (β-reduction) There exists $k \in \mathbb{N}$ constant such that for all t and $\underline{r}$,

$$\vdash_{k \cdot d(t)} \ (\ \lambda \underline{x}^\sigma . \, t[\underline{x}] \ ) \ \underline{r}^\sigma \ =_\tau \ t[\underline{r}] \ .$$

Proof: By straightforward induction on $d(t)$, using (2) when the induction step falls under (6). $\square$

Proposition 2.14 The following hold:

$$\begin{array}{rcl} d(\lambda \underline{x}.\, t) & \leq & 2 \cdot d(t) \\ S(\lambda \underline{x}.\, t) & \leq & 3 \cdot S(t) \\ mdg(\lambda \underline{x}.\, t) & \leq & max\{dg(\underline{x}) + 1 \, , \, mdg(t)\} + 1 \\ mar(\lambda \underline{x}.\, t) & \leq & max\{mar(t) + 1 \, , \, ar(\underline{x})\} + |\underline{x}| \end{array}$$

Proof: Structural induction on t, following Definition 2.12. $\square$

Remark 2.15 In order to increase readability we will omit the adornments of Σ, P and Π from now on. We consider that this side information can be figured out from the context in a straightforward way. On the other hand its display would only complicate the exposition.

3 A quantitative analysis of functional interpretation

Gödel's functional (*Dialectica*) interpretation, which assigns a formula $A^{\mathtt{D}} \equiv \exists \underline{x} \forall \underline{y} A_{\mathtt{D}}(\underline{x}; \underline{y}; \underline{a})$ with $A_{\mathtt{D}}$ quantifier-free and $\underline{x}, \underline{y}$ tuples of variables of finite type (s.t. $\{\underline{x}, \underline{y}, \underline{a}\} = \mathrm{V_f}(A_{\mathtt{D}})$) to each formula $A(\underline{a})$ (with $\{\underline{a}\} = \mathrm{V_f}(A)$), was first introduced in [21] and is also presented in [42](4), [53](3.5.1).

For notational simplicity we will omit tuples of free variables of formulas but for the places where their presence is relevant. We shall denote by

$$\begin{aligned} B(\underline{a'})^{\mathtt{D}} &\equiv \exists \underline{u} \forall \underline{v}\, B_{\mathtt{D}}(\underline{u}; \underline{v}; \underline{a'}) \\ C(\underline{a''})^{\mathtt{D}} &\equiv \exists \underline{g} \forall \underline{h}\, C_{\mathtt{D}}(\underline{g}; \underline{h}; \underline{a''}). \end{aligned}$$

The interpretation is given by the following list of rules:

Definition 3.1 (Gödel's functional interpretation)

$$\begin{aligned}
A^{\mathtt{D}} &:\equiv (A_{\mathtt{D}} :\equiv A) \text{ for prime formulas } A \\
(A \wedge B)^{\mathtt{D}} &:\equiv \exists \underline{x}, \underline{u} \forall \underline{y}, \underline{v}\, [(A \wedge B)_{\mathtt{D}} :\equiv A_{\mathtt{D}}(\underline{x}; \underline{y}) \wedge B_{\mathtt{D}}(\underline{u}; \underline{v})] \\
(\exists z A(\underline{a}, z))^{\mathtt{D}} &:\equiv \exists z, \underline{x} \forall \underline{y}\, [(\exists z A(\underline{a}, z))_{\mathtt{D}} :\equiv A_{\mathtt{D}}(\underline{x}; \underline{y}; \underline{a}, z)] \\
(\forall z A(\underline{a}, z))^{\mathtt{D}} &:\equiv \exists \underline{X} \forall z, \underline{y}\, (\forall z A(\underline{a}, z))_{\mathtt{D}} \qquad (7) \\
&\qquad\qquad (\forall z A(\underline{a}, z))_{\mathtt{D}} :\equiv A_{\mathtt{D}}(\underline{X}(z); \underline{y}; \underline{a}, z) \\
(A \to B)^{\mathtt{D}} &:\equiv \exists \underline{Y}, \underline{U} \forall \underline{x}, \underline{v}\, (A \to B)_{\mathtt{D}} \qquad (8) \\
&\qquad\qquad (A \to B)_{\mathtt{D}} :\equiv A_{\mathtt{D}}(\underline{x}; \underline{Y}(\underline{x}, \underline{v})) \to B_{\mathtt{D}}(\underline{U}(\underline{x}); \underline{v}) \\
(A \vee B)^{\mathtt{D}} &:\equiv \exists z^o, \underline{x}, \underline{u} \forall \underline{y}, \underline{v}\, (A \vee B)_{\mathtt{D}} \\
&\qquad (A \vee B)_{\mathtt{D}} :\equiv (z = 0 \to A_{\mathtt{D}}(\underline{x}; \underline{y})) \wedge (Iz1 = 0 \to B_{\mathtt{D}}(\underline{u}; \underline{v}))
\end{aligned}$$

Remark 3.2 For quantifier-free formulas A, $A^{\mathtt{D}} = A_{\mathtt{D}} = A$. The types and lengths of $\underline{x}$ and $\underline{y}$ depend only on the logical structure of A. Notice that $\mathrm{V_f}(A^{\mathtt{D}}) = \mathrm{V_f}(A)$ and $\mathrm{V_b}(A^{\mathtt{D}}) = \{\underline{x}, \underline{y}\}$.

Proposition 3.3 By induction on the structure of the formula A it can be easily proved that

$$qs(A^{\mathtt{D}}) = |\underline{x}, \underline{y}, \underline{a}| = qs(A) \tag{9}$$

□

Lemma 3.4 The following hold:

$$\begin{aligned} dg(\mathrm{V_b}(C^{\mathtt{D}})) &\leq vdg(C) + id(C) - k_0 + 1 && (10)\\ ar(\mathrm{V_b}(C^{\mathtt{D}})) &\leq var(C) + qs(C) \cdot (id(C) - k_0 + 1) && (11) \end{aligned}$$

Proof: The proof is by recursion on the structure of the formula C, following the Definition 3.1. We simply notice that

- $dg(\mathrm{V_b}(\cdot^D))$ may increase only at (8), with the quantity 1; (7) forces us to start with $1 + vdg(C)$, since $dg(X) = max\{dg(x)\,,\, dg(z) + 1\}$;
- $ar(\mathrm{V_b}(\cdot^D))$ may increase with the quantity 1 at (7) and with at most $|\underline{x}, \underline{v}| \leq qs(A \to B) \leq qs(C)$ at (8), hence

$$ar(\mathrm{V_b}(C^{\mathtt{D}})) \leq var(C) + fd(C) + qs(C) \cdot (id(C) - k_0)\ .$$

□

Definition 3.5 Let $\mathtt{Ax}$ be an arbitrary but fixed[21] set of axioms. For a set of closed terms $\mathfrak{Tm}$ and a set of formulas $\mathfrak{Fm}$ of $\mathtt{EIL}^{\omega}[\mathtt{Ax}]$ we define

- the *prerealization* relation by $PR[\mathfrak{Tm}, \mathfrak{Fm}] :\equiv \{(\underline{t}, A(\underline{a})) \subseteq \mathfrak{Tm}^{\leq\omega} \times \mathfrak{Fm} \mid |\underline{t}| = |\underline{x}|,\ \{\underline{a}\} = \mathrm{V_f}(A)$ and $\mathit{typ}(\underline{t}(\underline{a})) = \mathit{typ}(\underline{x})\}$. For $(\underline{t}, A(\underline{a})) \in PR[\mathfrak{Tm}, \mathfrak{Fm}]$ we abbreviate by $\{\!|\, \underline{t}\,,\, A \,|\!\} :\equiv \forall \underline{y}\, A_{\mathtt{D}}(\underline{t}(\underline{a}); \underline{y}; \underline{a})$.
- the *realization* relation by

$$RR[\mathfrak{Tm}, \mathfrak{Fm}] :\equiv \{\ (\underline{t}, A) \in PR[\mathfrak{Tm}, \mathfrak{Fm}]\ \mid\ \mathtt{EIL}^{\omega}[\mathtt{Ax}] \vdash \{\!|\underline{t}, A|\!\}\ \}$$

- the set of *realizing tuple selections* $RTS[\mathfrak{Fm}, \mathfrak{Tm}]$ as the set of those subsets of $RR[\mathfrak{Tm}, \mathfrak{Fm}]$ which are *functions* from $\mathfrak{Fm}$ to $\mathfrak{Tm}^{\leq\omega}$.

[21] See also Definition 3.10 and especially Remark 3.11.

We sometimes omit to specify Tm, Fm when they refer to all the closed terms, respectively formulas of $\mathtt{EIL}^{\omega}[\mathtt{Ax}]$. The set $\mathtt{Ax}$ will be determined by the context. Whenever $(\underline{t}, A) \in RR$ we denote this fact by $\underline{t}$ Dr A and say that

- $\underline{t}$ is a *realizing tuple* for $A^{\mathtt{D}}$;
- t is a *realizing term* for $A^{\mathtt{D}}$;
- $A^{\mathtt{D}}$ is *realized* by $\underline{t}$ or t.

We call

- *realizer* any realizing tuple or term;
- *realizer-free* a formula A for which $|\underline{x}| = 0$.

Definition 3.6 We say that a proof $\mathcal{P}$ is *realizer-free-normal* if all realizer-free formulas of $\mathcal{P}$ are located at the leaf level.

Remark 3.7 Let $\mathcal{P}$ be a realizer-free-normal proof. There exists no instance of $\mathtt{ER}_0$ in $\mathcal{P}$ since the conclusion is quantifier-free and consequently realizer-free. Realizer-free formulas of $\mathcal{P}$ may label only leaves of $\mathcal{P}$ which are left premises of $\mathtt{MP}$ instances. Indeed, if the conclusion in any of the rules $\mathtt{QR}$, $\mathtt{EXP}$, $\mathtt{IMP}$ is non-realizer-free then also the premise must be non-realizer-free. For the $\mathtt{MP}$ rule, if the conclusion is non-realizer-free then also the $A \to B$ premise must be non-realizer-free.

Definition 3.8 To any proof $\mathcal{P}$ in some extension of $\mathtt{EIL}^{\omega}$ we associate a realizer-free-normal proof $\mathcal{P}^{\mathrm{tr}}$ which is obtained from $\mathcal{P}$ by removing its maximal subtrees rooted at vertices labeled with realizer-free formulas, yet keeping these roots (which become assumptions in $\mathcal{P}^{\mathrm{tr}}$). There is a fairly simple algorithm which transforms $\mathcal{P}$ to $\mathcal{P}^{\mathrm{tr}}$ by recursion on proof structure.

Remark 3.9 The proofs we consider in the sequel are realizer-free-normal if not otherwise specified. See also Remark 3.32.

3.1 Axiom extensions of $\mathtt{EIL}^\omega$. The system $\mathtt{EIL}^\omega_+ + \mathtt{AC} + \mathtt{IP}_\forall + \mathtt{MK}$

Instances of the following three schemata are formulas whose correspondents under functional interpretation can be realized by very simple terms, basically projectors Π. This makes them the first to be considered for axiom extensions of $\mathtt{EIL}^\omega$ since by allowing them in proofs in the domain of functional interpretation causes no increase in complexity. Moreover the verifying proof is in $\mathtt{EIL}^\omega$ and has a constant bound on its depth. The first two are *logical axioms*, i.e. they are valid in classical logic; the third axiom is non-logical.

1. a variant of Markov's principle

   $\mathtt{MK}: \quad \neg\neg\,\exists \underline{x}\, A_0(\underline{x}) \to \exists \underline{x}\, \neg\neg\, A_0(\underline{x})$;

   the usual[22] formulation of Markov's principle

   $\mathtt{MK}': \quad \neg\neg\exists \underline{x} A_0(\underline{x}) \to \exists \underline{x} A_0(\underline{x})$

   can be deduced from $\mathtt{MK}$ with a proof which makes use of (5) and therefore has depth upper bounded by $k\, ld(A_0)$ for some $k \in \mathbb{N}$ constant; on the other hand the proof of $\mathtt{MK}$ from $\mathtt{MK}'$ has constant depth;

2. Independence of Premises for universal premises

   $\mathtt{IP}_\forall: \quad [\,\forall \underline{x}\, A_0(\underline{x}) \;\to\; \exists \underline{y}\, B(\underline{y})\,] \;\to\; \exists \underline{y}\, [\,\forall \underline{x}\, A_0(\underline{x}) \;\to\; B(\underline{y})\,]$,

   where $\underline{y} \notin \mathrm{V_f}(\forall \underline{x} A_0(\underline{x}))$;

3. the Axiom of Choice

   $\mathtt{AC}: \quad \forall \underline{x}\, \exists \underline{y}\, A(\underline{x}, \underline{y}) \;\to\; \exists \underline{Y}\, \forall \underline{x}\, A(\underline{x}, \underline{Y}(\underline{x}))$.

Another simple axiom extension of $\mathtt{EIL}^\omega$ is with realizer-free formulas since the quantitative analysis does not get affected in any way. There is a particular kind of such axiom extension which we consider in the sequel. Strictly, the terms t_1, t_2 which appear in prime formulas $t_1 = t_2$ of contractions $A \to A \wedge A$ and terms $\underline{s}$ involved in quantifier axioms $A(\underline{s}) \equiv \forall \underline{z} B(\underline{z}) \to B(\underline{s})$ or $A(\underline{s}) \equiv B(\underline{s}) \to \exists \underline{z} B(\underline{z})$ are part of the realizing term (see Section 3.3). However we do not count them in the quantitative analysis, but rather introduce new constants $\widetilde{t_1}, \widetilde{t_2}, \widetilde{s}$ associated to terms

[22] We prefer the variant $\mathtt{MK}$ because the verifying proof of its functional interpretation is much simpler than for $\mathtt{MK}'$. In the latter case the depth of the verifying proof is $k\, ld(A_0)$ for some $k \in \mathbb{N}$ constant.

t_1, t_2, s together with their defining axioms, such that any of the terms t_1, t_2, s contributes as much as a unit (plus the number of its free variables) of size to the realizing term. This is justified by the fact that we are merely interested in the complexity of functional interpretation itself; the terms t_1, t_2, s are not created by functional interpretation but are merely given as basic input data.

Definition 3.10 Let $\mathtt{Ax}$ be an arbitrary but fixed set of axioms and $\mathtt{Th}_{\mathtt{rf}}$ an arbitrary but fixed set of realizer-free theorems of $\mathtt{EIL}^{\omega}[\mathtt{Ax}]$. We define below two extensions $\mathtt{EIL}^{\omega}_{+}$ and $\mathtt{EIL}^{\omega}_{\mathtt{v}}$ of $\mathtt{EIL}^{\omega}[\mathtt{Ax}]$. The system $\mathtt{EIL}^{\omega}_{+}$ is obtained by simply adding $\mathtt{Th}_{\mathtt{rf}}$ to the set of axioms of $\mathtt{EIL}^{\omega}[\mathtt{Ax}]$. Let $\widetilde{\cdot}$ be a map which uniquely associates constants $\widetilde{t}$ to terms $t[\underline{a}]$ of $\mathtt{EIL}^{\omega}[\mathtt{Ax}]$ such that

$$\left.\begin{array}{rcl} dg(\widetilde{t}) & = & max\{dg(\underline{a})+1\,,\, dg(t)\} \\ ar(\widetilde{t}) & = & |\underline{a}| + ar(t) \end{array}\right\} \tag{12}$$

together with the defining axiom

$\mathtt{Ax}\widetilde{t}: \quad t[\underline{a}] = \widetilde{t}(\underline{a})$

Let Tm be an arbitrary but fixed set of $\mathtt{EIL}^{\omega}[\mathtt{Ax}]$ terms. The system $\mathtt{EIL}^{\omega}_{\mathtt{v}}$ is obtained by extending $\mathtt{EIL}^{\omega}_{+}$ with the defining axioms $\mathtt{Ax}\widetilde{t}$ for the newly introduced constants $\widetilde{t}$ associated to terms $t \in \mathrm{Tm}$ by (12).

Remark 3.11 All *arbitrary but fixed* items in the above definition are in fact context-dependent and will be in the subsequent concrete occurrences easy to figure out if not explicitly described.

3.2 The treatment of $\mathtt{EIL}^{\omega}$ rules

Remark 3.12 Recall that the formal proofs below are by default in $\mathtt{EIL}^{\omega}$. See Definition 3.5 for the meaning of the relations *PR*, *RR* and *RTS* below.

Lemma 3.13 The following hold for any proof $\mathcal{P}$:

$$qs(Vt(\mathcal{P})) = qs(Lv(\mathcal{P})) \quad \text{and} \quad ls(Vt(\mathcal{P})) = ls(Lv(\mathcal{P})) \tag{13}$$

$$\mathrm{V}(Vt(\mathcal{P})) = \mathrm{V}(Lv(\mathcal{P})) \quad \text{and} \quad \mathrm{C}(Vt(\mathcal{P})) = \mathrm{C}(Lv(\mathcal{P})) \tag{14}$$

Proof: The following (in)equalities are immediate:

$$qs(A \to \forall \underline{z} B(\underline{z})) = qs(A \to B(\underline{z})) \quad qs(\exists \underline{z} A(\underline{z}) \to B) = qs(A(\underline{z}) \to B)$$

$$qs(A \wedge B \to C) = qs(A \to B \to C) \qquad qs(B) \leq qs(A \to B)$$

It follows by structural induction on $\mathcal{P}$ that $qs(A) \leq qs(Lv(\mathcal{P}))$ for any formula $A \in Vt(\mathcal{P})$ and then $qs(Vt(\mathcal{P})) = qs(Lv(\mathcal{P}))$ is immediate. The argument for ls is identical and (14) has a similar proof, with $\subseteq$ instead of $\leq$. $\square$

Lemma 3.14 (MP) There exists $k \in \mathbb{N}$ constant and an algorithm which, given as input $(\underline{t_1}, A(\underline{a}))$, $(\underline{t_2}, \underline{t_3}, (A \to B)(\widetilde{\underline{a}})) \in PR$, produces as output $(\underline{t_4}, B(\underline{a'})) \in PR$ and the following hold (below $\{\underline{a_1}\} :\equiv \{\underline{a}\} - \{\underline{a'}\}$ and $\{\widetilde{\underline{a}}\} = \{\underline{a}\} \cup \{\underline{a'}\}$):

- t_4 is obtained from $t_4' :\equiv \Sigma(t_3, \underline{t_1}, \underline{a_1}) = \lambda \underline{a'}.\, t_3(\widetilde{\underline{a}}, \underline{t_1}(\underline{a}))$ by replacing the variables in $\underline{a_1}$ with constants O of appropriate type.
- $\{\!|\, \underline{t_1}\,,\, A \,|\!\}\ ,\ \{\!|\, \underline{t_2}, \underline{t_3}\,,\, A \to B \,|\!\} \vdash_k \{\!|\, \underline{t_4}\,,\, B \,|\!\}$
- $$\begin{aligned} d(\underline{t_4}) &\leq qs(A \to B) + d(\underline{t_1}, \underline{t_3}) && (15)\\ S(\underline{t_4}) &\leq 1 + qs(A \to B) \cdot S(\underline{t_1}, \underline{t_3}) && (16)\\ dg(\underline{t_4}) &\leq dg(\underline{t_3}) \quad \text{and} \quad ar(\underline{t_4}) \leq ar(\underline{t_3}) \\ mdg(\underline{t_4}) &\leq max\{mdg(\underline{t_1}, \underline{t_3})\,,\, dg(\underline{t_3}) + 1\} \\ mar(\underline{t_4}) &\leq max\{mar(\underline{t_1}, \underline{t_3})\,,\, ar(\underline{t_3}) + 1\,,\, ar(\underline{a_1})\} \end{aligned}$$

Proof: There exists $k \in \mathbb{N}$ constant such that for all $(\underline{t_1}, A)$, $(\underline{t_2}, \underline{t_3}, A \to B) \in PR$,

$$\underline{y} :\equiv \underline{t_2}(\widetilde{\underline{a}}, \underline{t_1}(\underline{a}), \underline{v})\ \frac{\forall \underline{y}\, A_{\mathtt{D}}(\underline{t_1}(\underline{a}); \underline{y})}{A_{\mathtt{D}}(\underline{t_1}(\underline{a}); \underline{t_2}(\widetilde{\underline{a}}, \underline{t_1}(\underline{a}), \underline{v}))}\, k$$

$$\underline{x} :\equiv \underline{t_1}(\underline{a})\ \frac{\forall \underline{x}, \underline{v}\, (A_{\mathtt{D}}(\underline{x}; \underline{t_2}(\widetilde{\underline{a}}, \underline{x}, \underline{v})) \to B_{\mathtt{D}}(\underline{t_3}(\widetilde{\underline{a}}, \underline{x}); \underline{v}))}{A_{\mathtt{D}}(\underline{t_1}(\underline{a}); \underline{t_2}(\widetilde{\underline{a}}, \underline{t_1}(\underline{a}), \underline{v})) \to B_{\mathtt{D}}(\underline{t_3}(\widetilde{\underline{a}}, \underline{t_1}(\underline{a})); \underline{v})}\, k$$

hence, by using MP once, we obtain that there exists $k \in \mathbb{N}$ constant such that for all $(\underline{t_1}, A), (\underline{t_2}, \underline{t_3}, A \to B) \in PR$,

$$\frac{\forall \underline{y}\, A_{\mathtt{D}}(\underline{t_1}(\underline{a}); \underline{y}) \qquad \forall \underline{x}, \underline{v}\, (A_{\mathtt{D}}(\underline{x}; \underline{t_2}(\widetilde{\underline{a}}, \underline{x}, \underline{v})) \to B_{\mathtt{D}}(\underline{t_3}(\widetilde{\underline{a}}, \underline{x}); \underline{v}))}{B_{\mathtt{D}}(\underline{t_3}(\widetilde{\underline{a}}, \underline{t_1}(\underline{a})); \underline{v})}\, k$$

Since by $\mathtt{Ax}\Sigma$ there exists $k \in \mathbb{N}$ constant such that for all $(\underline{t_1}, A)$, $(\underline{t_2}, \underline{t_3}, A \to B) \in PR$, $\vdash_k t_3(\widetilde{\underline{a}}, \underline{t_1}(\underline{a})) = t_4'(\underline{a'})$, and $B_{\mathtt{D}}$ is quantifier-free, by using (1) we obtain that there exists $k \in \mathbb{N}$ constant such that for all $(\underline{t_1}, A)$, $(\underline{t_2}, \underline{t_3}, A \to B) \in PR$,

$$B_{\mathtt{D}}(\underline{t_3}(\widetilde{\underline{a}}, \underline{t_1}(\underline{a})); \underline{v}) \vdash_k B_{\mathtt{D}}(\underline{t_4'}(\underline{a'}); \underline{v})$$

and we conclude that there exists $k \in \mathbb{N}$ constant such that for all $(\underline{t_1}, A), (\underline{t_2}, \underline{t_3}, A \to B) \in PR$,

$$\frac{\forall \underline{y}\, A_{\mathtt{D}}(\underline{t_1}(\underline{a}); \underline{y}) \qquad \forall \underline{x}, \underline{v}\, (A_{\mathtt{D}}(\underline{x}; \underline{t_2}(\widetilde{a}, \underline{x}, \underline{v})) \to B_{\mathtt{D}}(\underline{t_3}(\widetilde{a}, \underline{x}); \underline{v}))}{\forall \underline{v}\, B_{\mathtt{D}}(\underline{t_4'}(\underline{a'}); \underline{v})}\, k$$

We have that

$$\begin{array}{rcl} d(\underline{t_4}) & \leq & |\underline{t_1}, \underline{a_1}| + 1 + max\{d(\underline{t_1})\,,\, d(\underline{t_3})\} \\ S(\underline{t_4}) & \leq & 1 + (|\underline{t_1}, \underline{a_1}| + 1) \cdot max\{S(\underline{t_1})\,,\, S(\underline{t_3})\}\ . \end{array}$$

Inequalities (15) and (16) follow from

$$|\underline{t_1}, \underline{a_1}| + 1 \leq qs(A) + 1 \leq qs(A \to B)\,,$$

where for the last inequality we used that B is non-realizer-free. From

$$t_3(\widetilde{a}, \underline{t_1}(\underline{a})) = t_4'(\underline{a'}) = \Sigma(t_3, \underline{t_1}, \underline{a_1}, \underline{a'})$$

we obtain that

$$\begin{array}{rclcrcl} dg(\underline{t_4}) & \leq & dg(\underline{t_3}) & \qquad & dg(\underline{a_1}) & \leq & dg(\Sigma) = dg(t_3) + 1 \\ ar(\underline{t_4}) & \leq & ar(\underline{t_3}) & & ar(\Sigma) & = & ar(t_3) + 1 \end{array}$$

from which the remaining inequalities follow immediately. $\square$

Lemma 3.15 ($\mathtt{QR}\forall, \mathtt{QR}\exists$) There exists $k \in \mathbb{N}$ constant and an algorithm which, given as input $(\underline{t_1}, \underline{t_2}, A(\underline{a}) \to B(\underline{a'}, \underline{z})) \in PR$, produces as output $(\underline{t_3}, \underline{t_4}, A(\underline{a}) \to \forall \underline{z}\, B(\underline{a'}, \underline{z})) \in PR$ and the following hold ($\{\widetilde{a}\} :\equiv \{\underline{a}\} \cup \{\underline{a'}\}$):

- $t_3 :\equiv P\, t_1 = \lambda \widetilde{a}, \underline{x}, \underline{z}.\, t_1(\widetilde{a}, \underline{z}, \underline{x})$ and $t_4 :\equiv P\, t_2 = \lambda \widetilde{a}, \underline{x}, \underline{z}.\, t_2(\widetilde{a}, \underline{z}, \underline{x})$
- $\{\!|\, \underline{t_1}, \underline{t_2}\,,\ A \to B(\underline{z})\, |\!\}\ \vdash_k\ \{\!|\, \underline{t_3}, \underline{t_4}\,,\ A \to \forall \underline{z}\, B(\underline{z})\, |\!\}$
- $$\begin{array}{rcl} d(\underline{t_3}, \underline{t_4}) & \leq & d(\underline{t_1}, \underline{t_2}) + 1 \quad \text{and} \quad dg(\underline{t_3}, \underline{t_4}) = dg(\underline{t_1}, \underline{t_2}) \\ S(\underline{t_3}, \underline{t_4}) & \leq & S(\underline{t_1}, \underline{t_2}) + 1 \quad \text{and} \quad ar(\underline{t_3}, \underline{t_4}) = ar(\underline{t_1}, \underline{t_2}) \\ mdg(\underline{t_3}, \underline{t_4}) & \leq & max\{mdg(\underline{t_1}, \underline{t_2})\,,\, dg(\underline{t_1}, \underline{t_2}) + 1\} \\ mar(\underline{t_3}, \underline{t_4}) & \leq & max\{mdg(\underline{t_1}, \underline{t_2})\,,\, ar(\underline{t_1}, \underline{t_2}) + 1\} \end{array}$$

A corresponding statement holds for $\mathtt{QR}\exists$ as well, with the same bounds.

Proof: By definition,

$$\begin{aligned}(A \to B(\underline{a}', \underline{z}))^{\mathtt{D}} &\equiv \exists \underline{Y}, \underline{U} \forall \underline{x}, \underline{v}\, [A_{\mathtt{D}}(\underline{x}; \underline{Y}(\underline{x}, \underline{v})) \to B_{\mathtt{D}}(\underline{U}(\underline{x}); \underline{v}; \underline{a}', \underline{z})] \\ (A \to \forall \underline{z} B(\underline{a}', \underline{z}))^{\mathtt{D}} &\equiv \exists \underline{Y}, \underline{U} \forall \underline{x}, \underline{z}, \underline{v}\, [A_{\mathtt{D}}(\underline{x}; \underline{Y}(\underline{x}, \underline{z}, \underline{v})) \to B_{\mathtt{D}}(\underline{U}(\underline{x}, \underline{z}); \underline{v}; \underline{a}', \underline{z})]\,.\end{aligned}$$

By $\mathtt{Ax}P$, there exists $k \in \mathbb{N}$ constant such that for all $(\underline{t}_1, \underline{t}_2, A \to B(\underline{z})) \in PR$,

$$\vdash_k t_3(\widetilde{\underline{a}}, \underline{x}, \underline{z}, \underline{v}) = t_1(\widetilde{\underline{a}}, \underline{z}, \underline{x}, \underline{v}) \qquad \text{and} \qquad \vdash_k t_4(\widetilde{\underline{a}}, \underline{x}, \underline{z}) = t_2(\widetilde{\underline{a}}, \underline{z}, \underline{x})\ .$$

Since $A_{\mathtt{D}}(\underline{x}; \underline{y}) \to B_{\mathtt{D}}(\underline{u}; \underline{v}; \underline{a}', \underline{z})$ is quantifier-free, by using (1) we obtain that there exists $k \in \mathbb{N}$ constant such that for all $(\underline{t}_1, \underline{t}_2, A \to B(\underline{z})) \in PR$,

$$\frac{A_{\mathtt{D}}(\underline{x}; \underline{t}_1(\widetilde{\underline{a}}, \underline{z}, \underline{x}, \underline{v})) \to B_{\mathtt{D}}(\underline{t}_2(\widetilde{\underline{a}}, \underline{z}, \underline{x}); \underline{v}; \underline{a}', \underline{z})}{A_{\mathtt{D}}(\underline{x}; \underline{t}_3(\widetilde{\underline{a}}, \underline{x}, \underline{z}, \underline{v})) \to B_{\mathtt{D}}(\underline{t}_4(\widetilde{\underline{a}}, \underline{x}, \underline{z}); \underline{v}; \underline{a}', \underline{z})}\, k\ .$$

Further, there exists $k \in \mathbb{N}$ constant such that for all $(\underline{t}_1, \underline{t}_2, A \to B(\underline{z})) \in PR$,

$$\frac{\forall \underline{x}, \underline{v}\, (A_{\mathtt{D}}(\underline{x}; \underline{t}_1(\widetilde{\underline{a}}, \underline{z}, \underline{x}, \underline{v})) \to B_{\mathtt{D}}(\underline{t}_2(\widetilde{\underline{a}}, \underline{z}, \underline{x}); \underline{v}; \underline{a}', \underline{z}))}{\forall \underline{x}, \underline{z}, \underline{v}\, (A_{\mathtt{D}}(\underline{x}; \underline{t}_3(\widetilde{\underline{a}}, \underline{x}, \underline{z}, \underline{v})) \to B_{\mathtt{D}}(\underline{t}_4(\widetilde{\underline{a}}, \underline{x}, \underline{z}); \underline{v}; \underline{a}', \underline{z}))}\, k\ .$$

Obviously,

- $dg(t_3) = dg(t_1)$, $dg(t_4) = dg(t_2)$, hence $dg(\underline{t}_3, \underline{t}_4) = dg(\underline{t}_1, \underline{t}_2)$
- $ar(t_3) = ar(t_1)$, $ar(t_4) = ar(t_2)$, hence $ar(\underline{t}_3, \underline{t}_4) = ar(\underline{t}_1, \underline{t}_2)$

and the inequalities in the conclusion of this Lemma follow immediately. □

Lemma 3.16 (EXP, IMP) The following holds:

$$\{\!|\, \underline{t}_1, \underline{t}_2, \underline{t}_3\,,\ A \to (B \to C)\,|\!\} \quad = \quad \{\!|\, \underline{t}_1, \underline{t}_2, \underline{t}_3\,,\ A \wedge B \to C\,|\!\}.$$

Proof: By definition,

$$\begin{aligned}(A \wedge B \to C)^{\mathtt{D}} &\equiv \exists \underline{Y}, \underline{V}, \underline{G} \forall \underline{x}, \underline{u}, \underline{h} \\ &\qquad [A_{\mathtt{D}}(\underline{x}; \underline{Y}(\underline{x}, \underline{u}, \underline{h})) \wedge B_{\mathtt{D}}(\underline{u}; \underline{V}(\underline{x}, \underline{u}, \underline{h})) \to C_{\mathtt{D}}(\underline{G}(\underline{x}, \underline{u}); \underline{h})] \\ (A \to B \to C)^{\mathtt{D}} &\equiv \exists \underline{Y}, \underline{V}, \underline{G} \forall \underline{x}, \underline{u}, \underline{h} \\ &\qquad [A_{\mathtt{D}}(\underline{x}; \underline{Y}(\underline{x}, \underline{u}, \underline{h})) \to B_{\mathtt{D}}(\underline{u}; \underline{V}(\underline{x}, \underline{u}, \underline{h})) \to C_{\mathtt{D}}(\underline{G}(\underline{x}, \underline{u}); \underline{h})]\,.\end{aligned}$$

□

Theorem 3.17 There exists $k \in \mathbb{N}$ constant and an algorithm $\mathfrak{A}$ which does the following. Let $\mathcal{P}$ be some proof of a formula A in $\mathtt{EIL}^{\omega}_{+}$ and $\underline{s}_{(\cdot)} \in RTS[Lv(\mathcal{P})]$ a realizing tuple selection for the set of leaves of $\mathcal{P}$. Let $q_{\mathrm{o}} :\equiv max_{A \in Lv(\mathcal{P})}\, q(\underline{s}_A)$ for $q \in \{d, S, dg, ar, mdg, mar\}$ and $q_{\mathrm{o}} :\equiv q(Lv(\mathcal{P}))$ for $q \in \{qs, var\}$. Let[23] $\partial_{\mathtt{MP}} :\equiv \partial_{\mathtt{MP}}(\mathcal{P})$, $\partial_{\mathtt{QR}} :\equiv \partial_{\mathtt{QR}}(\mathcal{P})$ and $\partial :\equiv \partial(\mathcal{P})$. Let $\partial_{\mathrm{o}} \in \mathbb{N}$ be a number such that for all $A \in Lv(\mathcal{P})$, $\vdash_{\partial_{\mathrm{o}}} \{\!|\underline{s}_A, A|\!\}$. When $\mathfrak{A}$ is presented with $\mathcal{P}$ and $\underline{s}_{(\cdot)}$ at input, it produces as output $(\underline{t}, A) \in RR$ and the following hold:

$$d(\underline{t}) \quad \leq \quad d_{\mathrm{o}} + \partial_{\mathtt{QR}} + qs_{\mathrm{o}} \cdot (\partial_{\mathtt{MP}} - k_0) \tag{17}$$

$$S(\underline{t}) \quad \leq \quad (S_{\mathrm{o}} + \partial_{\mathtt{QR}} - k_0 + 1) \cdot qs_{\mathrm{o}}^{(\partial_{\mathtt{MP}} - k_{\mathrm{o}})} \tag{18}$$

$$dg(\underline{t}) \leq dg_{\mathrm{o}} \quad \text{and} \quad mdg(\underline{t}) \leq mdg_{\mathrm{o}} + 1 \tag{19}$$

$$ar(\underline{t}) \leq ar_{\mathrm{o}} \quad \text{and} \quad mar(\underline{t}) \leq max\{var_{\mathrm{o}}\,,\, mar_{\mathrm{o}} + 1\} \tag{20}$$

$$\mathtt{EIL}^{\omega}_{+} \quad \vdash_{\partial_{\mathrm{o}} + k\,\partial} \quad \{\!|\, \underline{t}\,,\, A \,|\!\} \tag{21}$$

Proof: The algorithm proceeds by recursion on the structure of $\mathcal{P}$, using the algorithms in Lemmas 3.14 and 3.15 as subprocedures at the $\mathtt{MP}$, respectively $\mathtt{QR}$ recursion steps; (21) follows immediately. We notice that dg and ar do not increase in the recursion, hence (19) and (20) are clear.

Let $\underline{e} \equiv e_1 \ldots e_n$ denote paths from some leaf to the root of $\mathcal{P}$, i.e., $(e_i)_{i \in \overline{1,n}}$ denote edges such that e_1 is incident with a leaf and e_n is incident with the root of $\mathcal{P}$. Let $(d^{\underline{e}}_i\,,\ S^{\underline{e}}_i)_{i \in \overline{0,n}}$ be a sequence of pairs of natural numbers defined by $(d^{\underline{e}}_0\,,\ S^{\underline{e}}_0) :\equiv (d_{\mathrm{o}}\,,\ S_{\mathrm{o}})$ and for $i \in \overline{1,n}$,

$$(d^{\underline{e}}_i\,,\ S^{\underline{e}}_i) \quad :\equiv \quad \begin{cases} (d^{\underline{e}}_{i-1} + qs_{\mathrm{o}}\,,\ qs_{\mathrm{o}} \cdot S^{\underline{e}}_{i-1} + 1), & \text{if } L(e_i) = \mathtt{MP} \\ (d^{\underline{e}}_{i-1} + 1\,,\ S^{\underline{e}}_{i-1} + 1) & \text{if } L(e_i) \in \mathtt{QR} \\ (d^{\underline{e}}_{i-1}\,,\ S^{\underline{e}}_{i-1}) & \text{otherwise} \end{cases} .$$

Using (13) it follows that $max_{\underline{e}}\, d^{\underline{e}}_n$ and $max_{\underline{e}}\, S^{\underline{e}}_n$ are upper bounds on d, S respectively. Inequalities (17) and (18) follow now immediately[24] . $\square$

Remark 3.18 Let us suppose that only unary (i.e., with $n = 1$) $\mathtt{ER}_{\mathtt{0}}$ is allowed in the verifying proof. The n-ary $\mathtt{ER}_{\mathtt{0}}$ can be obtained from unary $\mathtt{ER}_{\mathtt{0}}$ with a proof of depth proportional with n. It follows that we can upper bound the depths of proofs of lemmas used in verifying $\mathtt{MP}$ and $\mathtt{QR}$ with quantities proportional with qs_{o}. In consequence, (21) becomes

$$\mathtt{EIL}^{\omega}_{+} \vdash_{\partial_{\mathrm{o}} + k \cdot (qs_{\mathrm{o}} + \partial)} \{\!|\, \underline{t}\,,\, A \,|\!\} .$$

[23] See Section 1.2 for the meaning of $\partial_{\mathtt{MP}}(\mathcal{P})$, $\partial_{\mathtt{QR}}(\mathcal{P})$ and $\partial(\mathcal{P})$. Notice that $\mathtt{QR}\forall$, $\mathtt{QR}\exists$ and $\mathtt{MP}$ label edges in our $\mathtt{EIL}^{\omega}$-proof-trees $\mathcal{P}$ and $\mathtt{QR}$ cumulates both $\mathtt{QR}\forall$, $\mathtt{QR}\exists$ labels.

[24] At (18) an intermediate upper bound is $(S_{\mathrm{o}} + \partial_{\mathtt{QR}} - k_0) \cdot qs_{\mathrm{o}}^{(\partial_{\mathtt{MP}} - k_{\mathrm{o}})} + \sum_{i=0}^{(\partial_{\mathtt{MP}} - k_0) - 1} qs_{\mathrm{o}}^{\,i}$.

3.3 Bounds for realizing terms for $\mathtt{EIL}^{\omega}_{+}+\mathtt{AC}+\mathtt{IP}_{\forall}+\mathtt{MK}$ axioms

Remark 3.19 Recall that the formal proofs below are by default in $\mathtt{EIL}^{\omega}$.

Proposition 3.20 There exists $k \in \mathbb{N}$ constant such that for any instance A of $\mathtt{CT}\vee$, $\mathtt{WK}\vee$, $\mathtt{WK}\wedge$, $\mathtt{PM}\vee$, $\mathtt{PM}\wedge$, $\mathtt{SYL}$, $\mathtt{EPN}$, $\mathtt{EFQ}$, $\mathtt{TND}_0$, $\mathtt{MK}$, $\mathtt{IP}_{\forall}$, $\mathtt{AC}$, there exists a realizing tuple $\underline{t}$ for $A^{\mathtt{D}}$ such that

$$
\begin{aligned}
d(\underline{t}) &\leq k && (22)\\
S(\underline{t}) &\leq k && (23)\\
mdg(\underline{t}) &\leq k + vdg(A) + id(A) && (24)\\
mar(\underline{t}) &\leq k + var(A) + qs(A)\cdot(id(A) - k_0 + 2) && (25)\\
\mathtt{EIL}^{\omega} &\vdash_k \{\!|\, \underline{t}\,,\, A \,|\!\} && (26)
\end{aligned}
$$

Proof: We treat here $\mathtt{SYL}$ as an example, since it is the most complex among the above listed axioms. The remaining axioms are treated in the Appendix.

We have

$$
\begin{array}{l}
((A\to B)\wedge(B\to C)\to(A\to C))^{\mathtt{D}} \equiv \exists \underline{X},\underline{V},\underline{U'},\underline{H},\underline{Y'},\underline{G'}\ \forall \underline{y},\underline{u},\underline{v'},\underline{g},\underline{x'},\underline{h'}\\
\left(\begin{array}{c}
\left(\begin{array}{c} A_{\mathtt{D}}(\underline{X}(\underline{y},\underline{u},\underline{v'},\underline{g},\underline{x'},\underline{h'});\underline{y}(\underline{X}(\underline{y},\underline{u},\underline{v'},\underline{g},\underline{x'},\underline{h'}),\underline{V}(\underline{y},\underline{u},\underline{v'},\underline{g},\underline{x'},\underline{h'}))) \\ \to \\ B_{\mathtt{D}}(\underline{u}(\underline{X}(\underline{y},\underline{u},\underline{v'},\underline{g},\underline{x'},\underline{h'}));\underline{V}(\underline{y},\underline{u},\underline{v'},\underline{g},\underline{x'},\underline{h'})) \end{array}\right)\\
\bigwedge\\
\left(\begin{array}{c} B_{\mathtt{D}}(\underline{U'}(\underline{y},\underline{u},\underline{v'},\underline{g},\underline{x'},\underline{h'});\underline{v'}(\underline{U'}(\underline{y},\underline{u},\underline{v'},\underline{g},\underline{x'},\underline{h'}),\underline{H}(\underline{y},\underline{u},\underline{v'},\underline{g},\underline{x'},\underline{h'}))) \\ \to \\ C_{\mathtt{D}}(\underline{g}(\underline{U'}(\underline{y},\underline{u},\underline{v'},\underline{g},\underline{x'},\underline{h'}));\underline{H}(\underline{y},\underline{u},\underline{v'},\underline{g},\underline{x'},\underline{h'})) \end{array}\right)\\
\longrightarrow\\
\left(\begin{array}{c} A_{\mathtt{D}}(\underline{x'};\underline{Y'}(\underline{y},\underline{u},\underline{v'},\underline{g})(\underline{x'},\underline{h'})) \\ \to \\ C_{\mathtt{D}}(\underline{G'}(\underline{y},\underline{u},\underline{v'},\underline{g})(\underline{x'});\underline{h'}) \end{array}\right)
\end{array}\right)
\end{array}
$$

from

$$
\begin{array}{l}
((A\to B)\wedge(B\to C))^{\mathtt{D}} \equiv \exists \underline{Y},\underline{U},\underline{V'},\underline{G}\ \forall \underline{x},\underline{v},\underline{u'},\underline{h}\\
((A_{\mathtt{D}}(\underline{x};\underline{Y}(\underline{x},\underline{v}))\to B_{\mathtt{D}}(\underline{U}(\underline{x});\underline{v}))\wedge(B_{\mathtt{D}}(\underline{u'};\underline{V'}(\underline{u'},\underline{h}))\to C_{\mathtt{D}}(\underline{G}(\underline{u'});\underline{h})))
\end{array}
$$

from

$$\begin{aligned}
(A \to B)^{\mathtt{D}} &\equiv \exists \underline{Y}, \underline{U}\, \forall \underline{x}, \underline{v}\, (A_{\mathtt{D}}(\underline{x}; \underline{Y}(\underline{x}, \underline{v})) \to B_{\mathtt{D}}(\underline{U}(\underline{x}); \underline{v})) \\
(B \to C)^{\mathtt{D}} &\equiv \exists \underline{V'}, \underline{G}\, \forall \underline{u'}, \underline{h}\, (B_{\mathtt{D}}(\underline{u'}; \underline{V'}(\underline{u'}, \underline{h})) \to C_{\mathtt{D}}(\underline{G}(\underline{u'}); \underline{h})) \\
(A \to C)^{\mathtt{D}} &\equiv \exists \underline{Y'}, \underline{G'}\, \forall \underline{x'}, \underline{h'}\, (A_{\mathtt{D}}(\underline{x'}; \underline{Y'}(\underline{x'}, \underline{h'})) \to C_{\mathtt{D}}(\underline{G'}(\underline{x'}); \underline{h'}))
\end{aligned}$$

and we can take (here $\{\underline{a}\} = \mathrm{V_f}((A \to B) \wedge (B \to C) \to (A \to C))$)

$$\begin{aligned}
t_X &:\equiv \Pi = \lambda \underline{a}, \underline{y}, \underline{u}, \underline{v'}, \underline{g}, \underline{x'}, \underline{h'}.\, x' \\
t_H &:\equiv \Pi = \lambda \underline{a}, \underline{y}, \underline{u}, \underline{v'}, \underline{g}, \underline{x'}, \underline{h'}.\, h' \\
t_{U'} &:\equiv P\, \Sigma = \lambda \underline{a}, \underline{y}, \underline{u}, \underline{v'}, \underline{g}, \underline{x'}, \underline{h'}.\, u(\underline{x'}) \\
t_V &:\equiv P\, \Sigma = \lambda \underline{a}, \underline{y}, \underline{u}, \underline{v'}, \underline{g}, \underline{x'}.\, v'(\underline{u}(\underline{x'})) \\
t_{Y'} &:\equiv P\, (\Sigma\, \Sigma) = \lambda \underline{a}, \underline{y}, \underline{u}, \underline{v'}, \underline{g}, \underline{x'}, \underline{h'}.\, y(\underline{x'}, \underline{v'}(\underline{u}(\underline{x'}), \underline{h'})) \\
t_{G'} &:\equiv P\, \Sigma = \lambda \underline{a}, \underline{y}, \underline{u}, \underline{v'}, \underline{g}, \underline{x'}.\, g(\underline{u}(\underline{x'}))
\end{aligned}$$

The proofs of (22) and (23) are immediate, for (24) and (25) we use the results in Proposition 2.9 plus (10), respectively (9, 11) and for (26) we use $\mathtt{Ax}\Sigma$, $\mathtt{Ax}P$, $\mathtt{Ax}\Pi$, $\mathtt{Ax}D$ and (1). $\square$

Proposition 2.10 gives us an algorithm for associating terms $t_{A_{\mathtt{D}}}$ to formulas A such that $\vdash A_{\mathtt{D}} \leftrightarrow t_{A_{\mathtt{D}}} = 0$. Since $\mathrm{V}(t_{A_{\mathtt{D}}}) = \mathrm{V_f}(A_{\mathtt{D}})$ these $t_{A_{\mathtt{D}}}$ are generally not closed, whereas we want to produce closed realizing terms for contractions $A \to A \wedge A$. We could, of course, close these $t_{A_{\mathtt{D}}}$ via the λ-abstraction algorithm of Definition 2.12, however there is a more efficient way to achieve what we need.

Proposition 3.21 (Association of *closed* terms to *all* $\mathtt{EIL}^{\omega}_{+}$ formulas) There exists $k \in \mathbb{N}$ constant and an association of terms to $\mathtt{EIL}^{\omega}_{+}$ formulas $A \mapsto t^{\mathtt{D}}_A$ such that for all A (with $\{\underline{a}\} = \mathrm{V_f}(A)$)

$$\begin{aligned}
d(t^{\mathtt{D}}_A) &\leq k \cdot ld(A) && (27)\\
S(t^{\mathtt{D}}_A) &\leq k \cdot ls(A) && (28)\\
mdg(t^{\mathtt{D}}_A) &\leq k + vdg(A) + id(A) && (29)\\
mar(t^{\mathtt{D}}_A) &\leq k + var(A) + qs(A) \cdot (id(A) - k_0 + 2) && (30)\\
\mathtt{EIL}^{\omega}_{\mathtt{v}} &\vdash_{k \cdot ld(A)} A_{\mathtt{D}}(\underline{x}; \underline{y}; \underline{a}) \leftrightarrow t^{\mathtt{D}}_A(\underline{x}, \underline{y}, \underline{a}) = 0\ . && (31)
\end{aligned}$$

The $\widetilde{\cdot}$ constants in (31) are only those corresponding to terms occuring in A.

Proof: Induction on the structure of A. For prime formulas just take $t^{\mathtt{D}}_{\bot} :\equiv 1$ and (below $\{\underline{a_1}\} = \mathrm{V}(t_1)$, $\{\underline{a_2}\} = \mathrm{V}(t_2)$, and $\{\underline{a}\} = \mathrm{V_f}(t_1 = t_2)$)

$$t^{\mathtt{D}}_{t_1=t_2} \quad :\equiv \quad \Sigma\, E\, \widetilde{t_1}\, \widetilde{t_2} = \lambda \underline{a}.\, E\, \widetilde{t_1}(\underline{a_1})\, \widetilde{t_2}(\underline{a_2})$$

and otherwise define (below $\{\widetilde{\underline{a}}\} = \{\underline{a}\} \cup \{\underline{a'}\}$)

$$\begin{aligned}
t^{\mathtt{D}}_{A\wedge B} &:\equiv \Sigma\, \nu\, t^{\mathtt{D}}_A\, t^{\mathtt{D}}_B = \lambda \underline{x}, \underline{u}, \underline{y}, \underline{v}, \widetilde{\underline{a}}.\, \nu\, t^{\mathtt{D}}_A(\underline{x}, \underline{y}, \underline{a})\, t^{\mathtt{D}}_B(\underline{u}, \underline{v}, \underline{a'}) \\
t^{\mathtt{D}}_{\exists z A(\underline{a},z)} &:\equiv P\, t^{\mathtt{D}}_{A(\underline{a},z)} = \lambda z, \underline{x}, \underline{y}, \underline{a}.\, t^{\mathtt{D}}_{A(\underline{a},z)}(\underline{x}, \underline{y}, \underline{a}, z) \\
t^{\mathtt{D}}_{\forall z A(\underline{a},z)} &:\equiv \Sigma\, t^{\mathtt{D}}_{A(\underline{a},z)} = \lambda \underline{X}, z, \underline{y}, \underline{a}.\, t^{\mathtt{D}}_{A(\underline{a},z)}(\underline{X}(z), \underline{y}, \underline{a}, z) \\
t^{\mathtt{D}}_{A\to B} &:\equiv \Sigma\, \Sigma\, I\, t^{\mathtt{D}}_A\, t^{\mathtt{D}}_B = \lambda \underline{Y}, \underline{U}, \underline{x}, \underline{v}, \widetilde{\underline{a}}.\, I\, t^{\mathtt{D}}_A(\underline{x}, \underline{Y}(\underline{x}, \underline{v}), \underline{a})\, t^{\mathtt{D}}_B(\underline{U}(\underline{x}), \underline{v}, \underline{a'}) \\
t^{\mathtt{D}}_{A\vee B} &:\equiv \Sigma\, \Sigma\, \nu\, I\, t^{\mathtt{D}}_A\, t^{\mathtt{D}}_B\, I\, 1 = \\
&\qquad = \lambda z, \underline{x}, \underline{u}, \underline{y}, \underline{v}, \widetilde{\underline{a}}.\, \nu\, (I\, z\, t^{\mathtt{D}}_A(\underline{x}, \underline{y}, \underline{a}))\, (I\, (I\, z\, 1)\, t^{\mathtt{D}}_B(\underline{u}, \underline{v}, \underline{a'}))
\end{aligned}$$

The inequalities (27) and (28) are immediate, (29) and (30) follow from (10), respectively (9, 11) and (31) follows using the axioms $\mathtt{Ax}\Sigma$, $\mathtt{Ax}I$, $\mathtt{Ax}\nu$, $\mathtt{Ax}E$. □

Proposition 3.22 There exists $k \in \mathbb{N}$ constant such that for any instance A of $\mathtt{CT}\wedge$ there exists a realizing tuple $\underline{t}$ for $A^{\mathtt{D}}$ such that

$$\begin{aligned}
d(\underline{t}) &\leq k \cdot ld(A) \\
S(\underline{t}) &\leq k \cdot ls(A) \\
mdg(\underline{t}) &\leq k + vdg(A) + id(A) \\
mar(\underline{t}) &\leq k + var(A) + qs(A) \cdot (id(A) - k_0 + 3) \\
&\mathtt{EIL}^{\omega}_{\mathtt{v}} \vdash_{k\cdot ld(A)} \{\!|\, \underline{t}\,,\, A \,|\!\}\ . \qquad (32)
\end{aligned}$$

The $\widetilde{\cdot}$ constants in (32) are only those corresponding to terms occurring in A.

Proof: We have

$$\begin{aligned}
(A \equiv B \to B \wedge B)^{\mathtt{D}} \ \equiv\ & \exists \underline{Y}, \underline{X'}, \underline{X''}\, \forall \underline{x}, \underline{y'}, \underline{y''} \\
& [\, B_{\mathtt{D}}(\underline{x}; \underline{Y}(\underline{x}, \underline{y'}, \underline{y''})) \ \to\ B_{\mathtt{D}}(\underline{X'}(\underline{x}); \underline{y'}) \ \wedge\ B_{\mathtt{D}}(\underline{X''}(\underline{x}); \underline{y''}) \,]
\end{aligned}$$

and we can take (here $\{\underline{a}\} = \mathrm{V_f}(A) = \mathrm{V_f}(B)$)

$$\begin{aligned}
t_{X'} &:\equiv t_{X''} :\equiv \Pi = \lambda \underline{a}, \underline{x}.\, x \\
t_Y &:\equiv \Sigma\, D\, t^{\mathtt{D}}_B = \lambda \underline{a}, \underline{x}, \underline{y'}, \underline{y''}.\, D(t^{\mathtt{D}}_B(\underline{x}, \underline{y'}, \underline{a}), \underline{y''}, \underline{y'})
\end{aligned}$$

The proof of (32) is as follows:
By $\mathtt{Ax}D$, there exists $k \in \mathbb{N}$ constant such that for all B,

$$\begin{cases} \vdash_k \ t^{\mathtt{D}}_B(\underline{x}, \underline{y'}, \underline{a}) \ = \ 0 \ \rightarrow \ D(t^{\mathtt{D}}_B(\underline{x}, \underline{y'}, \underline{a}), \underline{y''}, \underline{y'}) = y'' \\ \vdash_k \ I\, t^{\mathtt{D}}_B(\underline{x}, \underline{y'}, \underline{a})\, 1 \ = 0 \ \rightarrow D(t^{\mathtt{D}}_B(\underline{x}, \underline{y'}, \underline{a}), \underline{y''}, \underline{y'}) = y' \end{cases}$$

and by using $\mathtt{ER}_{\mathtt{0}}$, there exists $k \in \mathbb{N}$ constant such that for all B,

$$\begin{cases} \vdash_k \ t^{\mathtt{D}}_B(\underline{x}, \underline{y'}, \underline{a}) = 0 \ \rightarrow \ B_{\mathtt{D}}(\underline{x}; \underline{D}(t^{\mathtt{D}}_B(\underline{x}, \underline{y'}, \underline{a}), \underline{y''}, \underline{y'})) \ \rightarrow \ B_{\mathtt{D}}(\underline{x}; \underline{y''}) \\ \vdash_k \ I\, t^{\mathtt{D}}_B(\underline{x}, \underline{y'}, \underline{a})\, 1 = 0 \ \rightarrow \ B_{\mathtt{D}}(\underline{x}; \underline{D}(t^{\mathtt{D}}_B(\underline{x}, \underline{y'}, \underline{a}), \underline{y''}, \underline{y'})) \ \rightarrow \ B_{\mathtt{D}}(\underline{x}; \underline{y'}) \ . \end{cases}$$

By $\mathtt{TND}_{\mathtt{0}}$ and $\mathtt{Ax}I$, there exists $k \in \mathbb{N}$ constant such that for all B,

$$\vdash_k \ t^{\mathtt{D}}_B(\underline{x}, \underline{y'}, \underline{a}) = 0 \ \vee \ I\, t^{\mathtt{D}}_B(\underline{x}, \underline{y'}, \underline{a})\, 1 \ = \ 0 \ .$$

¿From (31), there exists $k \in \mathbb{N}$ constant such that for all B,

$$\mathtt{EIL}^{\omega}_{\mathtt{v}} \ \vdash_{k \cdot ld(B)} \ B_{\mathtt{D}}(\underline{x}; \underline{y'}) \ \leftrightarrow \ t^{\mathtt{D}}_B(\underline{x}, \underline{y'}, \underline{a}) = 0,$$

hence there exists $k \in \mathbb{N}$ constant such that for all B,

$$\begin{cases} \mathtt{EIL}^{\omega}_{\mathtt{v}} \ \vdash_{k \cdot ld(B)} \ B_{\mathtt{D}}(\underline{x}; \underline{y'}) \rightarrow B_{\mathtt{D}}(\underline{x}; \underline{D}(t^{\mathtt{D}}_B(\underline{x}, \underline{y'}, \underline{a}), \underline{y''}, \underline{y'})) \rightarrow B_{\mathtt{D}}(\underline{x}; \underline{y''}) \\ \mathtt{EIL}^{\omega}_{\mathtt{v}} \ \vdash_{k \cdot ld(B)} \ \neg B_{\mathtt{D}}(\underline{x}; \underline{y'}) \rightarrow B_{\mathtt{D}}(\underline{x}; \underline{D}(t^{\mathtt{D}}_B(\underline{x}, \underline{y'}, \underline{a}), \underline{y''}, \underline{y'})) \rightarrow B_{\mathtt{D}}(\underline{x}; \underline{y'}) \ . \end{cases}$$

Since there exists $k \in \mathbb{N}$ constant such that for all A, B, C,

$$\frac{A \vee \neg A \, , \ A \rightarrow B \rightarrow C \, , \ \neg A \rightarrow B \rightarrow A}{B \rightarrow A \wedge C} \ k,$$

we finally obtain that there exists $k \in \mathbb{N}$ constant such that for all B,

$$\mathtt{EIL}^{\omega}_{\mathtt{v}} \ \vdash_{k \cdot ld(B)} \ B_{\mathtt{D}}(\underline{x}; \underline{D}(t^{\mathtt{D}}_B(\underline{x}, \underline{y'}, \underline{a}), \underline{y''}, \underline{y'})) \rightarrow B_{\mathtt{D}}(\underline{x}; \underline{y'}) \wedge B_{\mathtt{D}}(\underline{x}; \underline{y''}) \ .$$

Since there exists $k \in \mathbb{N}$ constant such that for all B,

$$\begin{aligned} &\vdash_k \ t_Y(\underline{a}, \underline{x}, \underline{y'}, \underline{y''}) = D(t^{\mathtt{D}}_B(\underline{x}, \underline{y'}, \underline{a}), \underline{y''}, \underline{y'}) \\ &\vdash_k \ t_{X'}(\underline{a}, \underline{x}) = x \\ &\vdash_k \ t_{X''}(\underline{a}, \underline{x}) = x \, , \end{aligned}$$

by (1) we obtain that there exists $k \in \mathbb{N}$ constant such that

$$\mathtt{EIL}^{\omega}_{\mathtt{v}} \ \vdash_{k \cdot ld(B)} \ B_{\mathtt{D}}(\underline{x}; \underline{t_Y}(\underline{a}, \underline{x}, \underline{y'}, \underline{y''})) \rightarrow B_{\mathtt{D}}(\underline{t_{X'}}(\underline{a}, \underline{x}); \underline{y''}) \wedge B_{\mathtt{D}}(\underline{t_{X''}}(\underline{a}, \underline{x}); \underline{y'})$$

which gives (32). The other inequalities follow immediately from Proposition 3.21. $\square$

Proposition 3.23 There exists $k \in \mathbb{N}$ constant such that for any instance $A(\underline{s})$ of $\mathtt{QA}\forall$, $\mathtt{QA}\exists$, there exists a realizing tuple $\underline{t}$ for $A^{\mathtt{D}}$ such that

$$\begin{array}{rcll} d(\underline{t}) & \leq & k + fd(A), \text{ when } A(\underline{s}) \in \mathtt{QA}\forall \text{ and} & (33)\\ d(\underline{t}) & \leq & k, \text{ when } A(\underline{s}) \in \mathtt{QA}\exists & \\ S(\underline{t}) & \leq & k + fd(A), \text{ when } A(\underline{s}) \in \mathtt{QA}\forall \text{ and} & (34)\\ S(\underline{t}) & \leq & k, \text{ when } A(\underline{s}) \in \mathtt{QA}\exists & \\ mdg(\underline{t}) & \leq & k + vdg(A) + id(A) & (35)\\ mar(\underline{t}) & \leq & k + var(A) + qs(A) \cdot (id(A) - k_0 + 2) & (36)\\ \mathtt{EIL}^{\omega}_{\mathtt{v}} & \vdash_k & \{\!| \, \underline{t}\,,\, A \, |\!\} & (37) \end{array}$$

The $\widetilde{\cdot}$ constants in (37) are only those corresponding to terms occurring in A.

Proof: Let $A(\underline{s}) \equiv \forall \underline{z} B(\underline{z}, \underline{a''}) \to B(\underline{s}, \underline{a''})$ be an instance of $\mathtt{QA}\forall$, s free for z in B. Let $\underline{a'} :\equiv \mathrm{V}(\underline{s})$ and $\underline{a} :\equiv \underline{a'}, \underline{a''} = \mathrm{V_f}(A(\underline{s}))$. Also $\underline{s} \equiv s_1, \ldots, s_n$ and let $\underline{a_i} :\equiv \mathrm{V}(s_i)$ for $i \in \overline{1,n}$. We have

$$(\forall \underline{z} B(\underline{z}) \to B(\underline{s}))^{\mathtt{D}} \equiv \exists \underline{Z}, \underline{Y}, \underline{X} \, \forall \underline{x}, \underline{y} \\ [\, B_{\mathtt{D}}(\underline{x}(\underline{Z}(\underline{x}, \underline{y})); \underline{Y}(\underline{x}, \underline{y}); \underline{Z}(\underline{x}, \underline{y})) \;\to\; B_{\mathtt{D}}(\underline{X}(\underline{x}); \underline{y}; \underline{s}) \,]$$

and we can take (recall from Definition 3.10 that $\widetilde{s_i}(\underline{a_i}) = s_i$)

$$\begin{array}{rcl} t_{Z_i} & :\equiv & \Sigma' \, \widetilde{s_i} = [\lambda u_i, \underline{a}, \underline{x}, \underline{y}.\, u_i(\underline{a_i})] \widetilde{s_i} = \lambda \underline{a}, \underline{x}, \underline{y}.\, \widetilde{s_i}(\underline{a_i}) \\ t_Y & :\equiv & \Pi = \lambda \underline{a}, \underline{x}, \underline{y}.\, y \\ t_X & :\equiv & P \, \Sigma \, \widetilde{\underline{s}} = [\lambda \underline{u}, \underline{a}, \underline{x}.\, x(u_1(\underline{a_1}), \ldots, u_n(\underline{a_n}))] \, \widetilde{\underline{s}} \\ & & \qquad = \lambda \underline{a}, \underline{x}.\, x(\widetilde{s_1}(\underline{a_1}), \ldots, \widetilde{s_n}(\underline{a_n})) \end{array}$$

¿From Proposition 2.9, $typ(u_i) = typ(\widetilde{s_i})$ and (12) it immediately follows that

$$\begin{array}{rcl} dg(\Sigma') & \leq & 2 + max\{dg(\underline{a}, \underline{x}, \underline{y}) \,,\, dg(s_i)\} \\ ar(\Sigma') & \leq & 1 + |\underline{a}, \underline{x}, \underline{y}| + ar(s_i) \\ dg(\Pi) & \leq & 1 + dg(\underline{a}, \underline{x}, \underline{y}) \\ ar(\Pi) & \leq & |\underline{a}, \underline{x}, \underline{y}| + ar(y) \\ dg(P) & \leq & 1 + dg(\Sigma) \leq 3 + max\{dg(\underline{a}, \underline{x}) \,,\, dg(\underline{s})\} \\ ar(P) & \leq & 1 + ar(\Sigma) \leq 2 + |\underline{a}, \underline{x}| + |\underline{s}| + ar(x) \end{array}$$

and (35), (36) now follow immediately from (10), respectively (9, 11), also using that $|\underline{z}| = |\underline{s}|$ and $\mathit{typ}(z_i) = \mathit{typ}(s_i)$.

The inequalities (33) and (34) are immediate from $|\underline{s}| \leq fd(A)$.

The proof of (37) uses the fact (which follows from (1)) that there exists $k \in \mathbb{N}$ constant such that for all $A(\underline{s})$,

$$\vdash_k B_{\mathtt{D}}(\, \underline{x}(\widetilde{s_1}(\underline{a_1})...\widetilde{s_n}(\underline{a_n}))\,;\underline{y};\, \widetilde{s_1}(\underline{a_1})...\widetilde{s_n}(\underline{a_n})\,) \to B_{\mathtt{D}}(\, \underline{x}(\widetilde{s_1}(\underline{a_1})...\widetilde{s_n}(\underline{a_n}))\,;\underline{y};\, \underline{s}\,)\,.$$

Let $A(\underline{s}) \equiv B(\underline{s}, \underline{a''}) \to \exists \underline{z} B(\underline{z}, \underline{a''})$ be an instance of $\mathtt{QA}\exists$, s free for z in B. The tuples $\underline{a'}$, $\underline{a}$ and $\underline{a_i}$ below are defined like in the $\mathtt{QA}\forall$ case above. We have

$$(B(\underline{s}) \to \exists \underline{z} B(\underline{z}))^{\mathtt{D}} \equiv \exists \underline{Y}, \underline{Z}, \underline{X}\, \forall \underline{x}, \underline{y}\, [\, B_{\mathtt{D}}(\underline{x}; \underline{Y}(\underline{x}, \underline{y}); \underline{s}) \;\to\; B_{\mathtt{D}}(\underline{X}(\underline{x}); \underline{y}; \underline{Z}(\underline{x}))\,]$$

and we can take (recall from Definition 3.10 that $\widetilde{s_i}(\underline{a_i}) = s_i$)

$$\begin{array}{rcl} t_Y & :\equiv & \Pi = \lambda \underline{a}, \underline{x}, \underline{y}.\, y \\ t_{Z_i} & :\equiv & \Sigma\, \widetilde{s_i} = (\lambda u_i, \underline{a}, \underline{x}.\, u_i(\underline{a_i}))\widetilde{s_i} = \lambda \underline{a}, \underline{x}.\, \widetilde{s_i}(\underline{a_i}) \\ t_X & :\equiv & \Pi = \lambda \underline{a}, \underline{x}.\, x \end{array}$$

The inequalities (33) and (34) are trivial, (35), (36) follow with an argument similar to the one in the $\mathtt{QA}\forall$ case. For (37) we use the fact (which follows from (1)) that there exists $k \in \mathbb{N}$ constant such that for all $A(\underline{s})$,

$$\vdash_k B_{\mathtt{D}}(\, \underline{x}\,;\, \underline{y}\,;\, s_1, \ldots, s_n\,) \to B_{\mathtt{D}}(\, \underline{x}\,;\, \underline{y}\,; \widetilde{s_1}(\,\underline{a_1}), \ldots, \widetilde{s_n}(\underline{a_n})\,)\;.$$

□

Notation 3.24 We will denote by $qs_{\mathrm{o}}(\mathcal{P}) :\equiv max\{2\,,\, qs(Lv(\mathcal{P}))\}$ and

$$\begin{array}{rclrcl} vdg_{\mathrm{o}}(\mathcal{P}) & :\equiv & vdg(Lv(\mathcal{P})) & var_{\mathrm{o}}(\mathcal{P}) & :\equiv & var(Lv(\mathcal{P})) \\ fd_1(\mathcal{P}) & :\equiv & fd(\mathtt{QA}\forall \cap Lv(\mathcal{P})) & id_{\mathrm{o}}(\mathcal{P}) & :\equiv & id(Lv(\mathcal{P})) \\ ld_1(\mathcal{P}) & :\equiv & ld(\mathtt{CT}\wedge \cap Lv(\mathcal{P})) & ls_1(\mathcal{P}) & :\equiv & ls(\mathtt{CT}\wedge \cap Lv(\mathcal{P})) \\ fid_{\mathrm{o}}(\mathcal{P}) & :\equiv & fid(Vt(\mathcal{P})) & ls_{\mathrm{o}}(\mathcal{P}) & :\equiv & ls(Lv(\mathcal{P})) \end{array}$$

We will omit $\mathcal{P}$ when this will be clear from the context.

Theorem 3.25 There exists $k \in \mathbb{N}$ constant such that for any proof $\mathcal{P}$ in $\mathtt{EIL}^{\omega}_{+}+\mathtt{AC}+\mathtt{IP}_{\forall}+\mathtt{MK}$ and any non-realizer-free $A \in Lv(\mathcal{P})$ there exists $\underline{t}_A$ such that $\underline{t}_A$ Dr A and the following hold:

- if A is not an instance of $(\mathtt{CT}\wedge, \mathtt{QA}\forall)$ then

$$\left.\begin{array}{rcl} d(\underline{t}_A) & \leq & k \\ S(\underline{t}_A) & \leq & k \\ mdg(\underline{t}_A) & \leq & k + vdg_{\mathrm{o}} + id_{\mathrm{o}} \\ mar(\underline{t}_A) & \leq & k + var_{\mathrm{o}} + qs_{\mathrm{o}} \cdot id_{\mathrm{o}} \\ \mathtt{EIL}^{\omega}_{\mathtt{v}} & \vdash_k & \{\!|\, \underline{t}_A\,,\, A \,|\!\} \end{array}\right\} \tag{38}$$

- if A is an instance of $\mathtt{CT}\wedge$, (38) holds except that

$$\begin{array}{rcl} d(\underline{t}_A) & \leq & k \cdot ld_{1} \\ S(\underline{t}_A) & \leq & k \cdot ls_{1} \\ \mathtt{EIL}^{\omega}_{\mathtt{v}} & \vdash_{k \cdot ld_{1}} & \{\!|\, \underline{t}_A\,,\, A \,|\!\} \end{array}$$

- if A is an instance of $\mathtt{QA}\forall$, (38) holds except that

$$\begin{array}{rcl} d(\underline{t}_A) & \leq & k + fd_{1} \\ S(\underline{t}_A) & \leq & k + fd_{1} \end{array}$$

The $\tilde{\cdot}$ constants of $\mathtt{EIL}^{\omega}_{\mathtt{v}}$ above[25] are only those required by the terms $\underline{t}_A$ and hence are limited to those corresponding to terms occurring in A.

Proof: Follows immediately from Propositions 3.20, 3.22, 3.23 and $k_0 \geq 10$. □

Theorem 3.26 There exists $k \in \mathbb{N}$ constant and an algorithm which does the following. Given as input a proof $\mathcal{P}$ of a formula A in $\mathtt{EIL}^{\omega}_{+}+\mathtt{AC}+\mathtt{IP}_{\forall}+\mathtt{MK}$ it produces as output $\underline{t}$ such that $\underline{t}$ Dr A and, with the notations 3.24 and abbreviations[26] $\partial_{\mathtt{MP}} :\equiv \partial_{\mathtt{MP}}(\mathcal{P})$, $\partial_{\mathtt{QR}} :\equiv \partial_{\mathtt{QR}}(\mathcal{P})$ and $\partial :\equiv \partial(\mathcal{P})$, the following hold:

$$d(\underline{t}) \leq k \cdot ld_{1} + \partial_{\mathtt{QR}} + qs_{\mathrm{o}} \cdot \partial_{\mathtt{MP}} \tag{39}$$

$$S(\underline{t}) \leq (k \cdot ls_{1} + \partial_{\mathtt{QR}}) \cdot qs_{\mathrm{o}}^{\partial_{\mathtt{MP}}} \tag{40}$$

$$mdg(\underline{t}) \leq k + vdg_{\mathrm{o}} + id_{\mathrm{o}} \tag{41}$$

$$mar(\underline{t}) \leq k + var_{\mathrm{o}} + qs_{\mathrm{o}} \cdot id_{\mathrm{o}} \tag{42}$$

$$\mathtt{EIL}^{\omega}_{\mathtt{v}} \vdash_{k \cdot (ld_{1} + \partial)} \{\!|\, \underline{t}\,,\, A \,|\!\} \tag{43}$$

[25] See also Definition 3.10 and Remark 3.11.

[26] See Footnote 23 for the meaning of $\partial_{\mathtt{MP}}(\mathcal{P})$, $\partial_{\mathtt{QR}}(\mathcal{P})$ and $\partial(\mathcal{P})$.

The $\widetilde{\cdot}$ constants of $\mathtt{EIL}^{\omega}_{\mathtt{v}}$ in (43) are among those corresponding to terms occurring in the leaves of $\mathcal{P}$.

Proof: Just a synthesis of the results in Theorems 3.17 and 3.25. For (39) and (40) we use that $fd_1 \leq qs_{\mathrm{o}}$ and $k_0 \geq 10$, hence

$$max\{k \cdot ld_1\,,\, k + fd_1\} + \partial_{\mathtt{QR}} + qs_{\mathrm{o}} \cdot (\partial_{\mathtt{MP}} - k_0) \leq k \cdot ld_1 + \partial_{\mathtt{QR}} + qs_{\mathrm{o}} \cdot \partial_{\mathtt{MP}}$$

$$(max\{k \cdot ls_1\,,\, k + fd_1\} + qs_{\mathrm{o}} + \partial_{\mathtt{QR}} - k_0 + 1) \cdot qs_{\mathrm{o}}^{(\partial_{\mathtt{MP}} - k_{\mathrm{o}})} \leq (k \cdot ls_1 + \partial_{\mathtt{QR}}) \cdot qs_{\mathrm{o}}^{\partial_{\mathtt{MP}}}$$

□

Notation 3.27 We will denote by

$$\begin{array}{rcl} wd_1(\mathcal{P}) & :\equiv & max\{wd(\mathtt{CT}\wedge \cap\, Lv(\mathcal{P}))\,,\, td(\mathtt{QA} \cap Lv(\mathcal{P}))\} \\ ws_1(\mathcal{P}) & :\equiv & max\{ws(\mathtt{CT}\wedge \cap\, Lv(\mathcal{P}))\,,\, ts(\mathtt{QA} \cap Lv(\mathcal{P}))\} \\ cdg_1(\mathcal{P}) & :\equiv & cdg((\mathtt{CT}\wedge \cup\, \mathtt{QA}) \cap Lv(\mathcal{P})) \\ car_1(\mathcal{P}) & :\equiv & car((\mathtt{CT}\wedge \cup\, \mathtt{QA}) \cap Lv(\mathcal{P})) \end{array}$$

We will omit $\mathcal{P}$ when this will be clear from the context.

Remark 3.28 Theorem 3.26 holds also when the terms t_1, t_2 which build prime formulas $t_1 = t_2$ of contractions $\mathtt{CT}\wedge$ and the quantifier axioms terms $\underline{s}$ are counted as components of the global realizer (instead of just taking the associated constants $\widetilde{t_1}, \widetilde{t_2}, \widetilde{s}$). We only need to use wd_1, ws_1 instead of ld_1, ls_1 and (41), (42) must be replaced with

$$\begin{array}{rcl} mdg(\underline{t}) & \leq & max\{k + vdg_{\mathrm{o}} + id_{\mathrm{o}}\,,\, cdg_1\} \\ mar(\underline{t}) & \leq & max\{k + var_{\mathrm{o}} + qs_{\mathrm{o}} \cdot id_{\mathrm{o}}\,,\, car_1\} \end{array}$$

Corollary 3.29 There exists $k' \in \mathbb{N}$ constant and an algorithm which does the following. Given as input a proof $\mathcal{P}$ of a formula $A \equiv \forall \underline{x}\, \exists \underline{y}\; B(\underline{x}, \underline{y})$ with $\{\underline{x}, \underline{y}\} = \mathrm{V_f}(B)$ in $\mathtt{EIL}^{\omega}_{+} + \mathtt{AC} + \mathtt{IP}_{\forall} + \mathtt{MK}$ it produces as output $\underline{t_Y}$ such that

$$\mathtt{EIL}^{\omega}_{\mathtt{v}} + \mathtt{AC} + \mathtt{IP}_{\forall} + \mathtt{MK} \vdash_{k' \cdot max\{ld_1 + \partial\,,\, ld(B)\}} \forall \underline{x}\; B(\underline{x}, \underline{t_Y}(\underline{x}))$$

Proof: In this case we have $A^{\mathtt{D}} \equiv \exists \underline{Y}, \underline{U}\, \forall \underline{x}, \underline{v}\; B_{\mathtt{D}}(\underline{U}(\underline{x}); \underline{v}; \underline{x}, \underline{Y}(\underline{x}))$, hence $\mathtt{EIL}^{\omega}_{\mathtt{v}} \vdash_{k \cdot (ld_1 + \partial)} \forall \underline{v}\; B_{\mathtt{D}}(\underline{t_U}(\underline{x}); \underline{v}; \underline{x}, \underline{t_Y}(\underline{x}))$ by Theorem 3.26 and further

$$\mathtt{EIL}^{\omega}_{\mathtt{v}} \vdash_{k \cdot (ld_1 + \partial)} \exists \underline{u}\, \forall \underline{v}\; B_{\mathtt{D}}(\underline{u}; \underline{v}; \underline{x}, \underline{t_Y}(\underline{x})) \quad [\equiv B^{\mathtt{D}}(\underline{x}, \underline{t_Y}(\underline{x}))\,] \qquad (44)$$

It can be easily proved by induction on $ld(B)$ that there exists $k'' \in \mathbb{N}$ such that for all formulas B,

$$\mathtt{EIL}^{\omega}+\mathtt{AC}+\mathtt{IP}_{\forall}+\mathtt{MK} \vdash_{k''\cdot ld(B)} B \leftrightarrow B^{\mathtt{D}} \tag{45}$$

The conclusion now follows immediately by combining (44) and (45). $\square$

Remark 3.30 If λ-abstraction were treated as primitive and Σ, P, Π were defined in terms of it then (40) would still hold. E.g., for Σ defined as $\lambda x, \underline{y}, \underline{z}.\, x(\underline{z_0}, y_1(\underline{z}^1), \underline{z_1}, \ldots, y_m(\underline{z}^m), \underline{z_m})$ we would have $S(\Sigma) \leq 2 \cdot |x, \underline{y}, \underline{z}|^2$ and on the other hand $|x, \underline{y}, \underline{z}| \leq qs_{\mathrm{o}}$ for all Σ which appear in $\underline{t}$. Similarly (40) would still hold if only Schönfinkel Σ and Π were allowed[27]. This follows from the λ-abstraction Definition 2.12. There exists $k \in \mathbb{N}$ constant such that at most $k \cdot |x, \underline{y}, \underline{z}|^2 \leq k \cdot {qs_{\mathrm{o}}}^2$ tuple-Schönfinkel Σ and Π are needed to simulate our Σ and any of these tuple-Schönfinkel Σ and Π can be defined[28] in terms of at most $k \cdot |x, \underline{y}, \underline{z}| \leq k \cdot qs_{\mathrm{o}}$ usual Schönfinkel Σ and Π.

Remark 3.31 If we allow only unary (see Remark 3.18) $\mathtt{ER}_{\mathtt{0}}$ in the verifying proof then (43) becomes

$$\mathtt{EIL}^{\omega}_{\mathtt{v}} \vdash_{k\cdot(ld_1+qs_{\mathrm{o}}+\partial)} \{\!|\, \underline{t}\,,\, A \,|\!\}$$

Remark 3.32 The algorithm of Theorem 3.26 can be applied to complete proofs $\mathcal{P}$ in $\mathtt{EIL}^{\omega}_{+}+\mathtt{AC}+\mathtt{IP}_{\forall}+\mathtt{MK}$ after a preprocessing phase to $\mathcal{P}^{\mathrm{tr}}$ via the procedure of Definition 3.8. Since $\mathtt{IEL}^{\omega} \vdash A \leftrightarrow A^{\mathtt{D}}$ for any realizer-free assumption A produced by the realizer-free-elimination procedure, the verifying proof can use the same assumptions as $\mathcal{P}^{\mathrm{tr}}$. A complete verifying proof in $\mathtt{EIL}^{\omega}_{\mathtt{v}}$ can be produced by (re)including the parts of $\mathcal{P}$ which were eliminated in the preprocessing phase.

3.4 Better bounds on the size of extracted terms

Smaller terms can be extracted if we use a simplification provided by the definitional equation of Σ. The size of the extracted terms becomes linear

[27] See Definition 2.4 for the notions of "tuple-Schönfinkel" and "Schönfinkel" combinators Σ. Also for "Schönfinkel" projectors Π.

[28] For Σ the proof is by induction on $|\underline{z}|$ of Definition 2.4. We have $\Sigma\, x\, y\, z\, \underline{z'} = x\, z\, \underline{z'}\, (y\, z\, \underline{z'}) = \Sigma'\, (xz)\, (yz)\, \underline{z'}$ hence $\Sigma = \lambda x, y, z.\, \Sigma'\, (xz)\, (yz)$. For Π we can use the iterated λ-abstraction $\lambda z_1.\, (\ldots \lambda z_n.\, z_i)$.

in the size of the proof at input. Nevertheless the use of extra Σ's brings an increase in type complexity. This can be avoided by using a more economical representation of the realizing tuples by means of pointers to parts which are shared by all members of a tuple. In such a setting all inequalities of Theorem 3.26 remain valid.

The simplification is based on the observation that all terms t_4 produced by MP (see Lemma 3.14) contain a common part. Namely $\underline{t_1}, \underline{O}$, which is somehow redundant to count for all t_4 in $\underline{t_4}$ - and this is what we have done by now. We give below a small example. Consider the following proof of C from A, $A \to B$ and $B \to C$:

$$\dfrac{\dfrac{A\,,\; A \to B}{B}\,,\quad B \to C}{C}$$

Let $\underline{t_1}$ Dr A, $(\underline{t_2}, \underline{t_3})$ Dr $(A \to B)$ and $(\underline{t_5}, \underline{t_6})$ Dr $(B \to C)$. The algorithm in Lemma 3.14 first produces $\underline{t_4}$ Dr B defined as $t_4 \equiv \Sigma(t_3, \underline{t_1}, \underline{O})$ and then produces the realizing tuple for C, namely $\underline{t_7}$ Dr C defined as

$$\begin{aligned} t_7 &\equiv \Sigma(t_6, \underline{t_4}, \underline{O'}) \\ &\equiv \Sigma(t_6, \Sigma(t_3^1, \underline{t_1}, \underline{O}), \ldots, \Sigma(t_3^{|t_3|}, \underline{t_1}, \underline{O}), \underline{O'}) \end{aligned}$$

We immediately notice that the tuple $\underline{t_1}, \underline{O}$ is common to all terms $t_4 \in \underline{t_4}$ and is multiply included in t_7. We describe below how it is possible to extract realizing terms such that the common parts which were previously multiply included are now counted only once for all the terms of a tuple.

Definition 3.33 For a proof $\mathcal{P}$ we define three *size* measures, denoted $S_i(\mathcal{P})$, $S_c(\mathcal{P})$, $S_m(\mathcal{P})$, to be used in the semi-**i**ntuitionistic (i.e., what we have already described), the **c**lassical and in the **m**onotone case respectively. The measure $S_m(\mathcal{P})$ will be used also for the time upper bounds (see Section 3.5) in all cases. All three size measures are obtained by adding the following to the sum of $qs(A \to B)$ for all MP-right-premises $A \to B$ plus the sum of $qs(C)$ for all QR-conclusions $qs(C)$ (below A are non-realizer-free leaves):

$S_i(\mathcal{P})$**:** the sum of $qs(A)$ for non-CT$\wedge$ A plus the sum of $ls(A)$ for CT$\wedge$ A;

$S_c(\mathcal{P})$**:** the sum of $ls(A)$ for all non-realizer-free leaves A;

$S_m(\mathcal{P})$**:** the sum of $qs(A)$ for all non-realizer-free leaves A.

Obviously $S_m(\mathcal{P}) \leq S_i(\mathcal{P}) \leq S_c(\mathcal{P})$.

Definition 3.34 For the tuples $\underline{t} \equiv t_1, \ldots, t_n$ extracted by the algorithm of Theorem 3.26 we define a *size* measure, denoted $Sz'(\underline{t})$ in the following way. There exists $m \geq 0$ and a tuple $\underline{t'}$ such that each $t_i \in \underline{t}$ is either of shape $t_i \equiv P_1^i(\ldots P_m^i(t^i))$ or of shape $t_i \equiv P_1^i(\ldots P_m^i(t^i(\underline{t'})))$ where $\{P_j^i\}_{j=1}^m$ and t^i are *characteristic* to t_i and $\underline{t'}$ is *common* to all t_i in the corresponding subset of $\underline{t}$. It is possible that $m = 0$ and/or the aforementioned subset is $\emptyset$. We define

$$Sz'(\underline{t}) :\equiv m \cdot |\underline{t}| \; + \; \Sigma_{t' \in \underline{t'}} \, S(t') \; + \; \Sigma_{i=1}^n \, S(t^i) \; .$$

Lemma 3.35 There exists $k \in \mathbb{N}$ constant such that for any term $P_1(P_2\, x)$ with P_1, P_2 permutations there exists a permutation P_3 such that $\vdash_k P_1(P_2\, x) = P_3\, x$.

Proof: By $\mathtt{Ax}P$ for P_1 we get $(P_1(P_2\, x))(\underline{z^p}) = P_2(x, \underline{z})$. We can now apply $\mathtt{Ax}P$ for P_2 and we distinguish two cases:

- $\underline{z} \equiv \underline{u^{p'}}, \underline{v}$ and $P_2(x, \underline{u^{p'}}) = x(\underline{u})$ hence $P_2(x, \underline{z}) = x(\underline{u}, \underline{v}) \equiv x(\underline{z^{p''}})$ and the last term is equal to $P_3(x, \underline{z^p})$ via a definitional equation for P_3.
- $\underline{z}, \underline{y} \equiv \underline{u^{p'}}$ and $P_2(x, \underline{u^{p'}}) = x(\underline{u})$ hence $(P_1(P_2\, x))(\underline{z^p}, \underline{y}) = x(\underline{u})$ and the last term is equal to $P_3(x, \underline{z^p}, \underline{y})$ via a definitional equation for P_3.

□

Lemma 3.36 There exists $k \in \mathbb{N}$ constant such that for any term $P_1(\ldots(P_m\, x))$ with $P_1, \ldots, P_m$ permutations there exists a permutation P_0 such that $\vdash_{k \cdot m} P_1(\ldots(P_m\, x)) = P_0\, x$.

Proof: Repeated applications of Lemma 3.35 and transitivity of equality. □

Theorem 3.37 There exists $k \in \mathbb{N}$ constant and an algorithm which does the following. Given as input a proof $\mathcal{P}$ of a formula A in $\mathtt{EIL}_+^\omega + \mathtt{AC} + \mathtt{IP}_\forall + \mathtt{MK}$ it produces as output $\underline{t}$ such that $\underline{t}$ Dr A with (43) and (below $\#_{\mathtt{MP}}$ denotes the number of $\mathtt{MP}$ instances in $\mathcal{P}$)

$$\begin{aligned} Sz'(\underline{t}) &\leq k \cdot S_i(\mathcal{P}) && (46)\\ 3\,\#_{\mathtt{MP}} &\leq d(\underline{t}) && (47)\\ \partial_{\mathtt{MP}} - k_0 &\leq mdg(\underline{t}) && (48)\\ 3\,\#_{\mathtt{MP}} &\leq mar(\underline{t}) && (49) \end{aligned}$$

Proof: The proof of (46) is by structural induction on $\mathcal{P}$. For axioms A we use the same realizing terms as before. When A is not an instance of $\mathtt{CT}\wedge$ or $\mathtt{QA}\forall$, (46) follows from $|\underline{t}| \leq qs(A)$. If $A \equiv B \to B \wedge B$ then we notice that $t^{\mathtt{D}}_B$ of Proposition 3.22 is common to all realizing t_Y, hence using (9) and (28),

$$Sz'(\underline{t_{X'}}, \underline{t_{X''}}, \underline{t_Y}) \leq k' \cdot |\underline{Y}, \underline{X'}, \underline{X''}| + S(t^{\mathtt{D}}_B) \leq k' \cdot qs(A) + k'' \cdot ls(A) \leq k \cdot ls(A) \ .$$

If $A \equiv \forall \underline{z} B(\underline{z}, \underline{a''}) \to B(\underline{s}, \underline{a''})$ then the tuple $\widetilde{\underline{s}}$ of Proposition 3.23 is common to all realizing t_X, hence

$$Sz'(\underline{t_Z}, \underline{t_Y}, \underline{t_X}) \leq k' \cdot |\underline{Z}, \underline{Y}, \underline{X}| + |\widetilde{\underline{s}}| \leq k \cdot qs(A) \ .$$

Nothing to prove for $\mathtt{EXP}$ and $\mathtt{IMP}$ instances, see Lemma 3.16.

For $\mathtt{QR}$ instances the proof is trivial using Lemma 3.15.

For $\mathtt{MP}$ instances we use Lemma 3.14 and further improve the result by applying a number of Σ definitional equations. The algorithm in Lemma 3.14 is presented with the tuples $\underline{t_3}$ and $\underline{t_1}$, represented [29] as

$$\left.\begin{array}{rcl} t_3 & \equiv & P'_1(\ldots P'_{m'}(t_0(\underline{t'}))) \quad = \quad P'(t_0(\underline{t'})) \\ t^i_1 & \equiv & P^i_1(\ldots P^i_m(t^i(\underline{t}))) \quad = \quad P_i(t^i(\underline{t})) \end{array}\right\} \text{Using Lemma 3.36}$$

and it produces (we assumed without loss of generality that $1 \leq m', m$)

$$\begin{array}{rcl} t_4 & \equiv & \Sigma_1(P'(t_0(\underline{t'})), P_1(t^1(\underline{t})), \ldots, P_n(t^n(\underline{t})), \underline{O}) = \\ & = & \Sigma_2(\Sigma_1, P', P_1, \ldots, P_n, t_0(\underline{t'}), t^1(\underline{t}), \ldots, t^n(\underline{t}), \underline{O}) = \\ & = & \Sigma_3(\Sigma_2, t_0, t^1, \ldots, t^n, \underline{t'}, \underline{t}, P', P_1, \ldots, P_n, \Sigma_1, \underline{O}) = \\ & = & P(\Sigma_3, \Sigma_2, P', t_0, t^1, \ldots, t^n, \underline{t'}, \underline{t}, P_1, \ldots, P_n, \Sigma_1, \underline{O}) \end{array}$$

hence we can actually take

$$t_4 \equiv (P\ \Sigma_3\ \Sigma_2\ P'\ t_0)(t^1, \ldots, t^n, \underline{t'}, \underline{t}, P_1, \ldots, P_n, \Sigma_1, \underline{O}) \tag{50}$$

[29] In the case when $\underline{t_3}$ or $\underline{t_1}$ comes from a (sub)proof which involved $\mathtt{CT}\wedge$ or $\mathtt{QA}\forall$ and no $\mathtt{MP}$ then we have an exception in the sense that only a part of the terms in the tuple share a common tuple, see also Definition 3.34. The reason should be obvious from the above treatment of $\mathtt{CT}\wedge$ and $\mathtt{QA}\forall$. The final shape of the term t_4 in (50) below is nevertheless not affected by this technical exception. After an $\mathtt{MP}$ all terms of the realizing tuple share a common tuple.

where $t^1, \ldots, t^n, \underline{t'}, \underline{t}, P_1, \ldots, P_n, \Sigma_1, \underline{O}$ is common to all $t_4 \in \underline{t_4}$. Hence

$$\begin{aligned} Sz'(\underline{t_4}) &\leq |P, \Sigma_3, \Sigma_2| \cdot |\underline{t_4}| + |\Sigma_1, \underline{O}| + Sz'(\underline{t_3}) + Sz'(\underline{t_1}) \\ &\leq 3 \cdot qs(A \to B) + Sz'(\underline{t_3}) + Sz'(\underline{t_1}), \end{aligned}$$

where for the last inequality we used that $|\underline{t_4}| + max\{1\,,\, |\underline{O}|\} \leq qs(A \to B)$.

In order to prove the remaining inequalities it is useful to denote by $\mathtt{cp}(\underline{t})$ the common tuple in the canonical representation of the tuple $\underline{t}$ (i.e., $\underline{t'}$). We have $|\mathtt{cp}(\underline{t_1})| + |\mathtt{cp}(\underline{t_3})| + 1 \leq |\mathtt{cp}(\underline{t_4})|$ because at least the constant Σ_1 appears new at each MP application. It follows that for the final extracted tuple $\underline{t}$ we have $\#_{\mathtt{MP}} \leq |\mathtt{cp}(\underline{t})|$. Now (47) and (49) are immediate because $|\mathtt{cp}(\underline{t})| \leq d(\underline{t})$ and $|\mathtt{cp}(\underline{t})| \leq mar(\underline{t})$. Also (48) is immediate once we notice that $dg(\mathtt{cp}(\underline{t}))$ increases by at least 1 at each MP application; this is due to the fact that t^i enters $\mathtt{cp}(\underline{t_4})$ and $dg(t^i) \geq dg(\underline{t}) + 1$.

The proof that (43) still holds is by straightforward computations. □

We notice that the price to pay for having smaller realizing terms is an increase in type complexity. This is unavoidable with the actual representation of terms. The maximal degree of the realizing term increases by at least 1 at each MP application. This is due to the fact that subterms from the private part, which have degree greater by at least 1 than the maximal degree of the common part now enter the new common part.

We can avoid the increase in type complexity only by modifying the term representation such that the terms in the common part are multiply pointed from each member of the realizing tuple. In this way Σ_3 is no longer needed for feeding the common part to each member of the realizing tuple. The increase in degree was due exactly to these Σ_3's. We can now state the following theorem, where Sz is defined in the new pointer setting similarly to Sz', i.e., counting common parts only once.

Theorem 3.38 There exists $k \in \mathbb{N}$ constant and an algorithm which does the following. Given as input a proof $\mathcal{P}$ of a formula A in $\mathtt{EIL}^{\omega}_{+} + \mathtt{AC} + \mathtt{IP}_{\forall} + \mathtt{MK}$ it produces as output $\underline{t}$ such that $\underline{t}$ Dr A, $Sz(\underline{t}) \leq k \cdot S_i(\mathcal{P})$ and the inequalities of Theorem 3.26 all hold. □

Remark 3.39 The following inequalities are immediate:

$$\begin{aligned} S(t) &\leq Sz(\underline{t}) \\ S_i(\mathcal{P}) &\leq (ls_{\scriptscriptstyle 1}(\mathcal{P}) + qs_{\scriptscriptstyle 0}(\mathcal{P})) \cdot 2^{\partial(\mathcal{P})} \leq 3 \cdot ls_{\scriptscriptstyle 1}(\mathcal{P}) \cdot qs_{\scriptscriptstyle 0}(\mathcal{P})^{\partial(\mathcal{P})}\ . \end{aligned}$$

They just express the fact that the new bounds on size are indeed better.

Remark 3.40 We will tacitly assume in the sequel that terms are represented with pointers in the manner described above.

3.5 Space and time complexity of the functional interpretation algorithm

In a real-world implementation of the algorithm of Theorem 3.26 we ought to count also the size of types associated to the $\texttt{EIL}^{\omega}$-constants as part of the size of the realizing terms. This *real-size* of the extracted terms actually gives also the time complexity of the algorithm[30] since what this does is only spelling down the extracted terms.

In order to compute the real-size we need to decide upon some representation of types. It turns out that the most efficient is to use *dags* [31]. We choose dags instead of normal binary trees[32] because dags allow the reuse of existent types via pointers. Hence given the input proof $\mathcal{P}$ we start with the types of all variables and constants which appear in $\mathcal{P}$ and build the types of constants which are produced by functional interpretation. We therefore need to count for the real-size only the number of new type-nodes which are created in order to represent the type of a newly created constant c. By straightforward computations it follows that there exists $k' \in \mathbb{N}$ constant such that for any formula C, the number of new type-nodes required by $\mathrm{V_b}(C^{\mathtt{D}})$ is at most $k' \cdot qs(C)^2 \cdot ls(C)$. Hence the number of new type-nodes required by the new variables created in the interpretation of the leaves, right $\texttt{MP}$–premises and $\texttt{QR}$–conclusions of $\mathcal{P}$ is at most $k' \cdot qs_{\mathrm{o}} \cdot ls_{\mathrm{o}} \cdot S_m(\mathcal{P})$. Then we can immediately see that whenever a new constant c of type $\underline{\sigma}\tau$ is created by the algorithm of Theorem 3.26, the types $\underline{\sigma}, \tau$ are immediately available from the existent terms or functional-interpretation-created variables. There exists $k'' \in \mathbb{N}$ constant such that for any such new constant c created at a leaf C, $\texttt{MP}$–instance of right–premise C or $\texttt{QR}$–instance of conclusion C of $\mathcal{P}$, at most $k'' \cdot |\underline{\sigma}|^2 \leq k'' \cdot qs(C)^2$ new type-nodes are necessary to represent the type of c. Hence overall we have at most $k'' \cdot qs_{\mathrm{o}} \cdot ls_{\mathrm{o}} \cdot S_m(\mathcal{P})$ newly created type-nodes in this category. We can now state the following theorem.

[30] The space complexity follows immediately by the principle that the space overhead of an algorithm is always less than its time overhead.

[31] Here "dag" is the usual acronym for "directed acyclic graph".

[32] The representation with binary trees is in fact equivalent to the usual parenthesized-strings representation.

Theorem 3.41 There exists $k \in \mathbb{N}$ constant such that the time overhead of the algorithm in Theorem 3.38 is upper bounded by $k \cdot qs_{\mathrm{o}} \cdot ls_{\mathrm{o}} \cdot S_m(\mathcal{P})$. □

4 Immediate extensions of the quantitative analysis

4.1 Treatment of classical $\mathtt{EIL}^{\omega}$. The system $\mathtt{ECL}^{\omega}_{+}+\mathtt{AC}_0$.

So far we have considered only semi-intuitionistic systems. We describe in the sequel how our complexity analysis can easily be adapted to classical logic (and theories) as well by applying it to the image of the classical system under a suitable negative translation. The so-called 'negative' or 'double-negation' translations have all in common the fact that the image of a formula is (intuitionistically equivalent to) a negative formula [33]. Negative translations were initially produced by Gödel [20], Gentzen, Kolmogorov, Glivenko. We use below a variant due to Kuroda of Gödel's translation which we further adapt such as to handle blocks of universal quantifiers.

Definition 4.1 (Kuroda's N-translation) To a formula A one associates $A^{\mathtt{N}} \equiv \neg\neg A^*$, where A^* is defined by structural induction on A as follows:

- $A^* :\equiv A$, if A is a prime formula;
- $(A \Box B)^* :\equiv A^* \Box B^*$, where $\Box \in \{\wedge, \vee, \rightarrow\}$
- $(\exists x A(x))^* :\equiv \exists x (A(x))^*$
- $(\forall \underline{x} A(\underline{x}))^* :\equiv \forall \underline{x} \neg\neg (A(\underline{x}))^*$, where $A(\underline{x}) \not\equiv \forall y B(y, \underline{x})$

Remark 4.2 $A^{\mathtt{N}}$ is realizer-free iff A is realizer-free.

[33] By definition, a formula is called *negative*, respectively *existential-free* if it is built up from negated prime, respectively prime formulas by means of $\bot$, $\wedge$, $\rightarrow$ and $\forall$ only. In our system negative formulas are trivially existential-free. On the other hand, $\mathtt{EIL}^{\omega} \vdash s =_o t \leftrightarrow \neg\neg(s =_o t)$ for any prime formula $s =_o t$ and hence also every existential-free formula is equivalent to a negative formula.

N-translation followed by functional interpretation gives a proof interpretation for theories based on classical logic [34]. Remark 4.2 implies that given a (complete) proof $\mathcal{P}$ in some classical system the following are equivalent:

- carry out the composed[35] interpretation to $\mathcal{P}^{\mathrm{tr}}$;
- first do the N-translation of $\mathcal{P}$, then apply the realizer-free-elimination algorithm of Definition 3.8 and finally carry out the functional interpretation of $(\mathcal{P}^{\mathtt{N}})^{\mathrm{tr}}$.

The former approach is obviously more efficient: one does not carry out the N-translation of parts which subsequently get eliminated.

Let $\mathtt{ECL}^{\omega}, \mathtt{ECL}^{\omega}_{+}$ be the classical versions[36] of $\mathtt{EIL}^{\omega}, \mathtt{EIL}^{\omega}_{+}$ respectively, obtained by replacing $\mathtt{TND}_0$ with the full tertium-non-datur schema $A \vee \neg A$. Let

$$\mathtt{AC}_0: \quad \forall \underline{x}\, \exists \underline{y}\, A_0(\underline{x}, \underline{y}) \to \exists \underline{Y}\, \forall \underline{x}\, A_0(\underline{x}, \underline{Y}(\underline{x}))$$

be the quantifier-free axiom-of-choice (with $\underline{x}$ and $\underline{y}$ of arbitrary types).

Remark 4.3 The proof–size measure S_c is introduced in Definition 3.33.

Proposition 4.4 There exists $k \in \mathbb{N}$ constant and an algorithm which does the following. Given as input a proof $\mathcal{P}$ of a formula A in $\mathtt{ECL}^{\omega}_{+}+\mathtt{AC}_0$ it produces as output a proof $\mathcal{P}^{\mathtt{N}}$ of $A^{\mathtt{N}}$ in $\mathtt{EIL}^{\omega}_{+}+\mathtt{AC}_0+\mathtt{MK}$ and the following hold:

- $\partial(\mathcal{P}^{\mathtt{N}}) \leq k \cdot \partial(\mathcal{P})$ and $S_i(\mathcal{P}^{\mathtt{N}}) \leq k \cdot S_c(\mathcal{P})$;
- $$\begin{aligned} qs_{\mathrm{o}}(\mathcal{P}^{\mathtt{N}}) \;\leq\; qs(Vt(\mathcal{P}^{\mathtt{N}})) \;\leq\; k \cdot qs(Vt(\mathcal{P})) &\overset{(13)}{=} k \cdot qs_{\mathrm{o}}(\mathcal{P}) \\ ld_{1}(\mathcal{P}^{\mathtt{N}}) \;\leq\; ls_{1}(\mathcal{P}^{\mathtt{N}}) \;\leq\; ls(Vt(\mathcal{P}^{\mathtt{N}})) \;\leq\; k \cdot ls(Vt(\mathcal{P})) &\overset{(13)}{=} k \cdot ls_{\mathrm{o}}(\mathcal{P}) \end{aligned}$$
- $id_{\mathrm{o}}(\mathcal{P}^{\mathtt{N}}) \leq k \cdot fid_{\mathrm{o}}(\mathcal{P})$; we must use $fid_{\mathrm{o}}(\mathcal{P})$ because in the N-translation a $\forall$ brings two $\neg$, hence in fact two $\to$ due to our treatment of negation;

[34] Details of the use of negative translation in combination with functional interpretation may be found, e.g., in [3, 27, 42].

[35] In fact parts which are produced by N-translation also need to be transformed.

[36] Below $\mathtt{EIL}^{\omega}_{+}$-based systems will appear for verifying the functional interpretation of proofs in $\mathtt{ECL}^{\omega}_{+}$-based systems. In virtue of Remark 4.2 it should be obvious that A is a realizer-free axiom from $\mathtt{Th}_{\mathtt{rf}}$ of $\mathtt{ECL}^{\omega}_{+}$ (see Definition 3.10) if and only if $A^{\mathtt{N}}$ is a realizer-free axiom from $\mathtt{Th}_{\mathtt{rf}}$ of $\mathtt{EIL}^{\omega}_{+}$.

- no new variable or constant appears in $\mathcal{P}^{\mathtt{N}}$, hence (using (14))

$$\begin{aligned} vdg(Vt(\mathcal{P}^{\mathtt{N}})) &\leq vdg(Vt(\mathcal{P})) = vdg(Lv(\mathcal{P})) \\ var(Vt(\mathcal{P}^{\mathtt{N}})) &\leq var(Vt(\mathcal{P})) = var(Lv(\mathcal{P})) \\ cdg(Vt(\mathcal{P}^{\mathtt{N}})) &\leq cdg(Vt(\mathcal{P})) = cdg(Lv(\mathcal{P})) \\ car(Vt(\mathcal{P}^{\mathtt{N}})) &\leq car(Vt(\mathcal{P})) = car(Lv(\mathcal{P})) \end{aligned}$$

Proof: The algorithm proceeds by recursion on the structure of $\mathcal{P}$, see [27] for details. The proof of its correctness makes use of the following schemata of intuitionistic logic:

$$\neg\neg(A \to B) \leftrightarrow (A \to \neg\neg B) \leftrightarrow (\neg\neg A \to \neg\neg B) \tag{51}$$

$$\neg\neg\forall \underline{x} \neg\neg A(\underline{x}) \leftrightarrow \forall \underline{x} \neg\neg A(\underline{x}) \tag{52}$$

$$A \to \neg\neg A \tag{53}$$

These schemata have proofs in which the axiom instances and intermediate formulas have size (depth) at most linear in the size (depth) of the formula to be proved. We only need to further notice that there exists $k' \in \mathbb{N}$ constant such that the following hold:

- the N-translation of any non-realizer-free axiom B of $\mathtt{ECL}^{\omega}_{+}+\mathtt{AC}_0$ is a theorem in $\mathtt{EIL}^{\omega}_{+}+\mathtt{AC}_0+\mathtt{MK}$ whose proof $\mathcal{P}'$ has the same structure for all instances of B, in particular the same depth; all formulas which appear in $\mathcal{P}'$ have size (depth) upper bounded by k' times the maximal size (depth) of B;

- any rule $A_1\,[,\,A_2] \vdash B$ of $\mathtt{ECL}^{\omega}_{+}+\mathtt{AC}_0$ is interpreted under N-translation to a proof $\mathcal{P}'$ of $B^{\mathtt{N}}$ from $A_1^{\mathtt{N}}\,[,\,A_2^{\mathtt{N}}]$; $\mathcal{P}'$ has the same structure for all instances of the rule, in particular the same depth; all formulas which appear in $\mathcal{P}'$ have size (depth) upper bounded by k' times the maximal size (depth) of $A_1\,[,\,A_2]$.

As an example we prove the above claim for $\mathtt{AC}_0$ and $\mathtt{QR}\forall$. The other axioms and rules are even easier.

Case $\mathtt{AC}_0$: We prove that there exists $k' \in \mathbb{N}$ constant such that for all A_0,

$$\mathtt{EIL}^{\omega}_{+}+\mathtt{AC}_0+\mathtt{MK} \vdash_{k'} [\,\forall \underline{x}\,\exists \underline{y}\,A_0(\underline{x},\underline{y}) \to \exists \underline{Y}\,\forall \underline{x}\,A_0(\underline{x},\underline{Y}(\underline{x}))\,]^{\mathtt{N}} \tag{54}$$

By (53), the conclusion of (54) is implied by

$$\forall \underline{x}\, \neg\neg\, \exists \underline{y}\, A_0(\underline{x}, \underline{y}) \to \exists \underline{Y}\, \forall \underline{x}\, \neg\neg\, A_0(\underline{x}, \underline{Y}(\underline{x}))\ .$$

This follows from $\mathtt{MK}$ and $\mathtt{AC_0}$ with a $\mathtt{IEL}^\omega$–proof of constant depth.

Case $\mathtt{QR}\forall$: $B \to A \vdash B \to \forall \underline{z} A$. By induction hypothesis we have a proof of $\neg\neg(B^* \to A^*)$. Then we use (51) and $\mathtt{MP}$ to get $B^* \to \neg\neg A^*$ and by $\mathtt{QR}\forall$, $B^* \to \forall \underline{z} \neg\neg A^*$. If $A \not\equiv \forall y C$ then $\forall \underline{z} \neg\neg A^* \equiv (\forall \underline{z} A)^*$. If $A \equiv \forall \underline{x} A'$ with $A' \not\equiv \forall y C$ then $A^* \equiv \forall \underline{x} \neg\neg A'^*$ and using (52) we get $B^* \to \forall \underline{z}, \underline{x} \neg\neg A'^*$ with $\forall \underline{z}, \underline{x} \neg\neg A'^* \equiv (\forall \underline{z} A)^*$. In any case we obtain $\neg\neg(B \to \forall \underline{z} A)^*$ (also using (53)). Hence overall the deduction of $(B \to \forall \underline{z} A)^{\mathtt{N}}$ from $(B \to A)^{\mathtt{N}}$ has constant depth. $\square$

Remark 4.5 The new quantifier axioms of $\mathcal{P}^{\mathtt{N}}$ are of shape $\forall \underline{z} B(\underline{z}) \to B(\underline{z})$ and these can be realized with simple projectors Π instead of the terms t_Z of Proposition 3.23.

Remark 4.6 Except for those triggered by $(A \to A \wedge A)^{\mathtt{N}}$, the contractions $\mathtt{CT}\wedge$ of $\mathcal{P}^{\mathtt{N}}$ are required by the N-translations of $A \vee \neg A$, $\mathtt{QA}\forall$ and $\mathtt{QR}\exists$. In the last two cases the verifying $\mathtt{CT}\wedge$ is brought by the critical implication

$$(\neg\neg A \to \neg\neg B) \to \neg\neg(A \to B) \tag{55}$$

of (51). The use of (55) can be avoided in the case of $\mathtt{IMP}$, $\mathtt{EXP}$ by using axiom versions of these rules[37], the non-critical converse of (55) and $\mathtt{MP}$.

Remark 4.7 $((\mathcal{P}^{\mathrm{tr}})^{\mathtt{N}})^{\mathrm{tr}} = (\mathcal{P}^{\mathtt{N}})^{\mathrm{tr}}$.

We are now able to describe an efficient algorithm for extracting realizing terms from (complete) proofs $\mathcal{P}$ in $\mathtt{ECL}^\omega_+ + \mathtt{AC_0}$. First $\mathcal{P}$ is transformed to $\mathcal{P}^{\mathrm{tr}}$ and then to $(\mathcal{P}^{\mathrm{tr}})^{\mathtt{N}}$ via the algorithm of Theorem 4.4. In a second phase $(\mathcal{P}^{\mathrm{tr}})^{\mathtt{N}}$ is transformed[38] to $((\mathcal{P}^{\mathrm{tr}})^{\mathtt{N}})^{\mathrm{tr}}$ and the algorithm of Theorem 3.26 is applied to it. Using Proposition 4.4, Theorems 3.38 and 3.41, Notation 3.24 and the abbreviations $\partial :\equiv \partial(\mathcal{P})$, $S_c :\equiv S_c(\mathcal{P})$ and $S_m :\equiv S_m(\mathcal{P})$ we can state the following theorem.

[37] The axiom versions of $\mathtt{IMP}$ and $\mathtt{EXP}$ are simply realized with projectors Π. This follows immediately from the fact that $(\, A \to (B \to C)\,)^{\mathtt{D}} \equiv (\, A \wedge B \to C\,)^{\mathtt{D}}$. See also Lemma 3.16 and the comment before the cases $\mathtt{MK}, \mathtt{IP}_\forall, \mathtt{AC}$ in the proof of Proposition A.1.

[38] Here only the parts produced by N-translation need to be transformed.

Theorem 4.8 There exists $k \in \mathbb{N}$ constant and an algorithm which does the following. Given as input a proof $\mathcal{P}$ of a formula A in $\mathtt{ECL}^{\omega}_{+}+\mathtt{AC}_{\mathtt{0}}$ it produces as output $\underline{t}$ such that $\underline{t}$ Dr $A^{\mathtt{N}}$ and the following hold:

$$d(\underline{t}) \leq k \cdot (ls_{\mathrm{o}} + qs_{\mathrm{o}} \cdot \partial) \tag{56}$$

$$S(\underline{t}) \leq Sz(\underline{t}) \leq k \cdot S_c \leq k \cdot (ls_{\mathrm{o}} + \partial) \cdot (k \cdot qs_{\mathrm{o}})^{k \cdot \partial} \tag{57}$$

$$mdg(\underline{t}) \leq vdg_{\mathrm{o}} + k \cdot fid_{\mathrm{o}} \tag{58}$$

$$mar(\underline{t}) \leq var_{\mathrm{o}} + k \cdot qs_{\mathrm{o}} \cdot fid_{\mathrm{o}} \tag{59}$$

$$\mathtt{EIL}^{\omega}_{\mathtt{v}} \vdash_{k \cdot (ls_{\mathrm{o}} + \partial)} \{\!|\, \underline{t}\,,\, A^{\mathtt{N}} \,|\!\}\,. \tag{60}$$

The time overhead of the algorithm is upper bounded by $k \cdot qs_{\mathrm{o}} \cdot ls_{\mathrm{o}} \cdot S_m$. The $\widetilde{\cdot}$ constants of $\mathtt{EIL}^{\omega}_{\mathtt{v}}$ in (60) are among those corresponding to terms occurring in the leaves of $\mathcal{P}^{\mathtt{N}}$. $\square$

Remark 4.9 In the above theorem we use the more general quantity ∂ instead of the more detailed ones $\partial_{\mathtt{QR}}$ and $\partial_{\mathtt{MP}}$ which appear in Theorem 3.26. We do so because the N-translations of $\mathtt{QR}\forall$, $\mathtt{QR}\exists$, $\mathtt{EXP}$ and $\mathtt{IMP}$ trigger new $\mathtt{MP}$ instances in $\mathcal{P}^{\mathtt{N}}$. Hence $\partial_{\mathtt{MP}}(\mathcal{P}^{\mathtt{N}}) \geq \partial(\mathcal{P})$ already.

Corollary 4.10 Let $A \equiv \forall \underline{x}\, \exists \underline{y}\, A_0(\underline{x}, \underline{y})$ with A_0 quantifier-free and $\mathrm{V_f}(A_0) = \{\underline{x}, \underline{y}\}$. The theorem above holds also with A instead of $A^{\mathtt{N}}$, i.e., $\underline{t}$ Dr A with $\mathtt{EIL}^{\omega}_{\mathtt{v}} \vdash_{k \cdot (ls_{\mathrm{o}} + \partial)} \forall \underline{x}\, A_0(\underline{x}, \underline{t}(\underline{x}))$ and (56), (57), (58), (59).

Proof: There exists $k' \in \mathbb{N}$ constant such that, using (52) and (5),

$$\mathtt{EIL}^{\omega}_{+}+\mathtt{MK} \vdash_{k' \cdot ld(A_0)} (\forall \underline{x} \exists \underline{y} A_0(\underline{x}, \underline{y}))^{\mathtt{N}} \rightarrow \forall \underline{x} \exists \underline{y} A_0(\underline{x}, \underline{y})\ .$$

¿From (13) it follows that the quantity $ld(A_0)$ gets absorbed into ls_{o}. $\square$

4.2 A quantitative analysis of monotone functional interpretation

The second author realized in [26] that a much simpler extraction procedure applies if the goal is to extract majorizing functionals $\underline{t^*}$ for the realizing terms $\underline{t}$ of $A^{\mathtt{D}}$, i.e., terms $\underline{t^*}$ such that

$$\mathtt{M}: \qquad \exists \underline{x}\, [\underline{t}^*\ maj\ \underline{x}\ \wedge\ \forall \underline{a}, \underline{y}\, A_{\mathtt{D}}(\underline{x}(\underline{a}), \underline{y}, \underline{a})] \qquad .$$

Here maj is W.A. Howard's majorization relation (see [24]) and

$\underline{y}\ maj\ \underline{x} :\equiv\ \wedge(y\ maj\ x)$. This is of significance since $\underline{t^*}$ suffices for many (if not most) applications of functional interpretation. These range from conservation results (e.g., for weak König's lemma [30]) to the proof mining of concrete proofs [28, 31, 32, 35]. We noticed in Section 3 that the contraction $A \to A \wedge A$ is by far the most complicated axiom in the usual functional interpretation. Monotone functional interpretation features a very simple treatment of $A \to A \wedge A$ and therefore the extraction process for $\underline{t^*}$ becomes much simpler than the one for $\underline{t}$.

Definition 4.11 Let $\mathtt{EIL}^\omega_\mathtt{M}$ be an extension of $\mathtt{EIL}^\omega$ with the following:

- An inequality relation $\geq_o$ for type-o-objects with the usual axioms plus

  $$1 \geq_o I\, x^o\, y^o \quad 1 \geq_o \nu\, x^o\, y^o \quad 1 \geq_o E\, x^o\, y^o\ .$$

  Inequality for higher types is defined extensionally by

  $$x \geq_{\underline{\sigma o}} y \quad :\equiv \quad \forall \underline{z^\sigma}\,(\ x\,\underline{z}\ \geq_o\ x\,\underline{z}\ )\ .$$

  The *majorization* relation is defined by $maj_o :\equiv\ \geq_o$ and

  $$x^*\ maj_{\underline{\sigma o}}\ x \quad :\equiv \quad \forall \underline{z^\sigma}, \underline{y^\sigma}\,(\ \underline{z}\ maj_{\underline{\sigma}}\ \underline{y}\ \to\ x^*\,\underline{z}\ maj_o\ x\,\underline{y}\ )\ ,$$

  where $\underline{z}\ maj_{\underline{\sigma}}\ \underline{y}$ is an abbreviation for $\wedge_{\sigma \in \underline{\sigma}}(z\ maj_\sigma\ y)$.

- A *maximum* constant M_o of type ooo with the axioms

  $\mathtt{Ax}M:\qquad M_o\,x\,y \geq_o x \qquad M_o\,x\,y \geq_o y \qquad M_o\ maj\ M_o\ .$

  Maximum constants for higher types are defined by

  $$M_{\underline{\sigma o}} :\equiv \Sigma\, M_o = \lambda x^{\underline{\sigma o}}, y^{\underline{\sigma o}}, \underline{z^\sigma}.\, M_o\,(x\,\underline{z})\,(y\,\underline{z})\ .$$

- A schema of explicit definability for arbitrary quantifier-free formulas:

  $$\mathtt{ED}[A_0]:\quad \exists Y\, \forall \underline{a}\ [\,(1 \geq_o Y(\underline{a})) \wedge (A_0(\underline{a}) \leftrightarrow Y(\underline{a}) =_o 0)\,]\ .$$

- Axioms $S\ maj\ S$ and $O\ maj\ O$.

Remark 4.12 In the presence of a minimal amount of arithmetic $S\ maj\ S$ and $O\ maj\ O$ are immediately provable. Also the constants $\geq_o$, ν, I, E and M_o can be defined such that the remaining axioms of Definition 4.11 become provable (see also Remark 2.5).

Remark 4.13 The formulas $\Sigma\ maj\ \Sigma$, $\Pi\ maj\ \Pi$ and $P\ maj\ P$ hold in $\mathtt{EIL}^{\omega}_{\mathtt{M}}$ with proofs of depths proportional with the arities of Σ, Π and P respectively. Then $M_\rho\ maj\ M_\rho$ holds for arbitrary ρ with a formal proof of depth proportional with $ar(\rho)+1$.

Lemma 4.14 There exists $k \in \mathbb{N}$ constant such that for any tuple of terms $\underline{s}$ of $\mathtt{EIL}^{\omega}_{\mathtt{M}}$ (with $\mathrm{V}(\underline{s}) = \{\underline{x}\}$) there exist corresponding terms $\underline{s^*}$ of $\mathtt{EIL}^{\omega}_{\mathtt{M}}$ (with $\mathrm{V}(\underline{s^*}) = \{\underline{x^*}\}$) such that

$$\mathtt{EIL}^{\omega}_{\mathtt{M}} \vdash \underline{x^*}\ maj\ \underline{x} \to \underline{s^*}\ maj\ \underline{s}\ . \tag{61}$$

Proof: The constants O and S trivially majorize themselves by the last clause of Definition 4.11. On the other hand, $\Sigma\, M = \lambda z, \underline{x}, \underline{x'}.\, M\, x\, x'$ majorizes D and $\Pi\, 1 = \lambda x^o, y^o.\, 1$ majorizes I, ν and E. Using Remark 4.13 we have that Σ, Π, P and M majorize themselves. The conclusion follows immediately by induction on $d(\underline{s})$. $\square$

Corollary 4.15 Let $\widetilde{s}, \widetilde{s^*}$ be constants associated to terms s, s^* like in Definition 3.10. ¿From (61) it immediately follows that

$$\vdash \underline{\widetilde{s^*}}\ maj\ \underline{\widetilde{s}} \tag{62}$$

Definition 4.16 We denote by $\mathtt{EIL}^{\omega}_{\mathtt{M},+}$ the system $(\mathtt{EIL}^{\omega}_{\mathtt{M}})_+$ where "+" includes all formulas $\underline{\widetilde{s^*}}\ maj\ \underline{\widetilde{s}}$ as axioms. We take them as axioms because we consider that the (formal) proof in (62) is not created by monotone functional interpretation. Also let $\mathtt{EIL}^{\omega}_{\mathtt{M},\mathtt{v}}$ be the corresponding $(\mathtt{EIL}^{\omega}_{\mathtt{M}})_{\mathtt{v}}$.

In [26] realizing terms are presented for the monotone functional interpretation of all axioms of $\mathtt{EIL}^{\omega}_{\mathtt{M}}+\mathtt{AC}+\mathtt{IP}_\forall+\mathtt{MK}$. They are the same as for the usual functional interpretation, except that

- $A \to A \wedge A$ is realized by terms $\Sigma M = \lambda \underline{a}, \underline{x}, \underline{y'}, \underline{y''}.\, M\, y'\, y''$ and $\Pi = \lambda \underline{a}, \underline{x}.\, x$; compare this with the results of Proposition 3.22;
- $A \vee A \to A$ is realized by $\Sigma M = \lambda \underline{a}, z, \underline{x}, \underline{x'}.\, M\, x\, x'$ and Π;
- $A \vee B \to B \vee A$ is realized by terms Π and $\Pi\, 1$;
- the schema $\mathtt{ED}$ itself is trivially realized by $\Pi\, 1 = \lambda \underline{a}.\, 1$;

- $\forall \underline{z} A(\underline{z}) \to A(\underline{s})$ is realized by terms obtained from the realizing terms of the usual functional interpretation by replacing the constants $\widetilde{s}$ with the corresponding $\widetilde{s^*}$ where $\underline{s^*}$ are given by Lemma 4.14.

Using Remark 4.13 it follows that there exists $k \in \mathbb{N}$ constant such that the verifying proof for some axiom A of $\mathtt{EIL}^{\omega}_{\mathtt{M}}+\mathtt{AC}+\mathtt{IP}_{\forall}+\mathtt{MK}$ has depth upper bounded by $k \cdot qs(A)$. The verifying proof for $\mathtt{CT}\wedge$ makes use of $\mathtt{ED}$.

Remark 4.17 The proof–size measure S_m is introduced in Definition 3.33. The proof–depth measures $\partial_{\mathtt{MP}}$, $\partial_{\mathtt{QR}}$ and ∂ are introduced in Section 1.2. In the following theorem we will abbreviate by $\partial_{\mathtt{MP}} :\equiv \partial_{\mathtt{MP}}(\mathcal{P})$, $\partial_{\mathtt{QR}} :\equiv \partial_{\mathtt{QR}}(\mathcal{P})$, $\partial :\equiv \partial(\mathcal{P})$ and $S_m :\equiv S_m(\mathcal{P})$.

Since monotone functional interpretation uses the same algorithm as the usual functional interpretation for producing realizing terms for conclusions given the realizing terms for premises, the following analogue of Theorem 3.38 holds.

Theorem 4.18 There exists $k \in \mathbb{N}$ constant and an algorithm which does the following. Given as input a proof $\mathcal{P}$ of A in $\mathtt{EIL}^{\omega}_{\mathtt{M},+}+\mathtt{AC}+\mathtt{IP}_{\forall}+\mathtt{MK}$ it produces as output $\underline{t^*}$ such that, with the notations 3.24, the following hold:

$$\begin{array}{rcl} d(\underline{t}) & \leq & k + \partial_{\mathtt{QR}} + qs_{\mathrm{o}} \cdot \partial_{\mathtt{MP}} \\ S(\underline{t}) \leq Sz(\underline{t}) & \leq & min\{k \cdot S_m \, , \, k \cdot \partial_{\mathtt{QR}} \cdot qs_{\mathrm{o}}^{\partial_{\mathtt{MP}}}\} \leq k \cdot qs_{\mathrm{o}}^{\partial} \qquad (63) \\ mdg(\underline{t}) & \leq & k + vdg_{\mathrm{o}} + id_{\mathrm{o}} \\ mar(\underline{t}) & \leq & k + var_{\mathrm{o}} + qs_{\mathrm{o}} \cdot id_{\mathrm{o}} \\ \mathtt{EIL}^{\omega}_{\mathtt{M},\mathtt{v}} \vdash_{k\cdot(qs_{\mathrm{o}}+\partial)} \exists \underline{x} \, (\underline{t^*} \; maj \; \underline{x} \wedge \forall \underline{a}, \underline{y} \, A_{\mathtt{D}}(\underline{x}(\underline{a}), \underline{y}, \underline{a})) & & (64) \end{array}$$

The time overhead of the algorithm is upper bounded by $k \cdot qs_{\mathrm{o}} \cdot ls_{\mathrm{o}} \cdot S_m$. The $\widetilde{\cdot}$ constants of $\mathtt{EIL}^{\omega}_{\mathtt{M},\mathtt{v}}$ in (64) are among those corresponding to terms occurring in the leaves of $\mathcal{P}$.

Proof: The rightmost inequality of (63) follows from a suitable adaptation of Remark 3.39 to the monotone case. We now only need to comment on (64). In order to build the verifying proof for $\mathtt{MP}$ we need to use the following lemma:

$$(\underline{y_1} \; maj \; \underline{x_1}) \wedge (\underline{y_3} \; maj \; \underline{x_3}) \to \wedge [\, \Sigma(y_3, \underline{y_1}, \underline{O}) \; maj \; \Sigma(x_3, \underline{x_1}, \underline{O}) \,] \qquad (65)$$

Using Remark 4.13 it follows that there exists $k' \in \mathbb{N}$ such that for all its instances, lemma (65) has a proof of depth at most $k' \cdot |y_3, \underline{y_1}, \underline{O}|$. When used for verifying $\mathtt{MP}$, $|y_3, \underline{y_1}, \underline{O}| \leq qs_{\mathrm{o}}$ hence (64) follows immediately. $\square$

Remark 4.19 If (65) were taken as axiom, the depth of verifying $\mathtt{MP}$ would be upper bounded by a constant, just like in the case of usual functional interpretation. On the other hand (65), $\Sigma\ maj\ \Sigma$, $\Pi\ maj\ \Pi$, $P\ maj\ P$ and $M\ maj\ M$ would have constant-depth proofs in $\mathtt{EIL}^{\omega}_{\mathtt{M}}$ if the underlying logical system handled tuples of conjunctions more smoothly. In such a case (64) could be replaced with

$$\mathtt{EIL}^{\omega}_{\mathtt{M},\mathtt{v}} \vdash_{k\cdot\partial} \exists \underline{x}\,(\underline{t^*}\ maj\ \underline{x} \wedge \forall \underline{a},\underline{y}\, A_{\mathtt{D}}(\underline{x}(\underline{a}),\underline{y},\underline{a}))$$

Hence the bound on verifying proof depth would be better than in the usual functional interpretation case, see (43). The smoother treatment of tuples of conjunctions would actually be normal in our context with free use of tuples in both quantifier axioms/rules and the extensionality rule $\mathtt{ER}_{\mathtt{0}}$.

Let $\mathtt{ECL}^{\omega}_{\mathtt{M},+}$ be the classical variant of $\mathtt{EIL}^{\omega}_{\mathtt{M},+}$. Combined with N-translation, monotone functional interpretation carries over to $\mathtt{ECL}^{\omega}_{\mathtt{M},+}+\mathtt{AC}_{\mathtt{0}}$ and the upper bounds on size and proof depth are smaller than in the functional interpretation case. The following analogue of Theorem 4.8 + Corollary 4.10 holds.

Theorem 4.20 There exists $k \in \mathbb{N}$ constant and an algorithm which does the following. Given as input a proof $\mathcal{P}$ of A in $\mathtt{ECL}^{\omega}_{\mathtt{M},+}+\mathtt{AC}_{\mathtt{0}}$, it produces as output $\underline{t^*}$ such that, with the notations 3.24 and the abbreviations $\partial :\equiv \partial(\mathcal{P})$ and $S_m :\equiv S_m(\mathcal{P})$ the following hold:

$$\begin{aligned} d(\underline{t^*}) &\leq k\cdot qs_{\mathrm{o}}\cdot\partial \\ S(\underline{t^*}) &\leq Sz(\underline{t^*}) \leq k\cdot S_m \leq (k\cdot qs_{\mathrm{o}})^{k\cdot\partial} \\ mdg(\underline{t^*}) &\leq vdg_{\mathrm{o}} + k\cdot fid_{\mathrm{o}} \\ mar(\underline{t^*}) &\leq var_{\mathrm{o}} + k\cdot qs_{\mathrm{o}}\cdot fid_{\mathrm{o}} \\ \mathtt{EIL}^{\omega}_{\mathtt{M},\mathtt{v}} \vdash_{k\cdot(qs_{\mathrm{o}}+\partial)} &\exists \underline{x}\,[\,\underline{t^*}\ maj\ \underline{x} \wedge \forall \underline{a},\underline{y}\,(A^{\mathtt{N}})_{\mathtt{D}}(\underline{x}(\underline{a}),\underline{y},\underline{a})\,] \end{aligned} \tag{66}$$

For $A \equiv \forall \underline{x}\,\exists \underline{y}\, A_0(\underline{x},\underline{y})$ with A_0 quantifier-free and $\{\underline{x},\underline{y}\} = \mathrm{V_f}(A_0)$, (66) can be replaced with

$$\mathtt{EIL}^{\omega}_{\mathtt{M},\mathtt{v}} \vdash_{k\cdot(ld(A_0)+qs_{\mathrm{o}}+\partial)} \exists \underline{Y}\,[\,\underline{t^*}\ maj\ \underline{Y} \wedge \forall \underline{x}\, A_0(\underline{x},\underline{Y}(\underline{x}))\,] \tag{67}$$

The time overhead of the algorithm is upper bounded by $k\cdot qs_{\mathrm{o}}\cdot ls_{\mathrm{o}}\cdot S_m$. The $\widetilde{\ }$ constants of $\mathtt{EIL}^{\omega}_{\mathtt{M},\mathtt{v}}$ in (66, 67) are among those corresponding to terms occurring in the leaves of $\mathcal{P}^{\mathtt{N}}$. $\square$

In concrete applications of monotone functional interpretation, $\mathtt{EIL}^{\omega}_{\mathtt{M}}$ will be extended by certain arithmetical (and even analytical) principles (see Section 5 below).

In the presence of a modest amount of arithmetic we can make use of t^* extracted by monotone functional interpretation in the following way. Let $\underline{x}$, $\underline{y}$ be of type o. Then (67) implies $\forall \underline{x} \exists \underline{y} \leq \underline{t^*}(\underline{x})\, A_0(\underline{x}, \underline{y})$ and therefore, using bounded search applied to $\underline{t^*}$ and a characteristic term t_{A_0} for A_0 one easily constructs $\underline{t}$ such that $\forall \underline{x} A_0(\underline{x}, \underline{t}(\underline{x}))$. This also works for $\underline{x}$ of type 1 using the construction $x^{\mathtt{M}}(i) :\equiv max_{j \leq i}\, x(j)$ since $x^{\mathtt{M}}\ maj\ x$. Moreover, for sentences of the form $\forall x^1\, \forall z \leq_1 s\, \exists y^o\, A_0(x, z, y)$ with s closed term one can easily obtain a type-2-term $\widehat{t}$ from t^* such that $\vdash\ \forall x^1\, \forall z \leq_1 s\, \exists y \leq_o \widehat{t}(x)\, A_0(x, z, y)$ by taking $\widehat{t}(x) :\equiv t^*(x^M, s^*)$ where s^* is a majorizing term for s. The term $\widehat{t}$ provides a uniform bound on y which is independent from z. See [30] for more details. This feature of monotone functional interpretation is of crucial importance in applications to numerical analysis [31, 32, 26, 28, 35, 38] where $\{z \mid z \leq_1 s\}$ is used to represent compact Polish spaces. Since $A_0(x, z, y)$ is monotone (i.e., $A_0(x, z, y_1) \wedge y_2 \geq y_1 \rightarrow A_0(x, z, y_2)$) in most applications, the term $\widehat{t}$ will not only be a bound but actually a realizer for $\exists y$.

Hence in this context monotone functional interpretation even provides a realizer which is independent from z and of simpler structure than realizers produced by the usual functional interpretation (see [26] for more on this).

5 Extensions to Arithmetic and fragments of Analysis

Both Gödel's functional interpretation and the monotone functional interpretation apply to intuitionistic and, via the negative translation, also classical arithmetic [21, 33, 53] (even in finite types) and fragments thereof [10, 33, 45].

5.1 Treatment of Primitive Recursive Arithmetic $\mathtt{PRA}^{\omega}$

Let us first consider Feferman's system [16] $\mathtt{PRA}^{\omega}$ (and its intuitionistic variant $\mathtt{PRA}^{\omega}_{\mathtt{i}}$) of primitive recursive arithmetic in all finite types, where only quantifier-free induction and ordinary Kleene-primitive recursive functionals are included.

Definition 5.1 Let $\mathtt{PRA}^{\omega}_{\mathtt{i}}$ be an extension of $\mathtt{EIL}^{\omega}$ with the following:

- Kleene recursor[39] constants $\widehat{R}_\rho$ with axioms

$$\mathtt{Ax}\widehat{R}: \quad \left\{ \begin{array}{rcl} \widehat{R}_\rho(0,y,z,\underline{v}) & =_o & z(\underline{v}) \\ \widehat{R}_\rho(Sx,y,z,\underline{v}) & =_o & y(\widehat{R}_\rho(x,y,z,\underline{v}),x,\underline{v}) \end{array} \right. .$$

- Axioms (the usual primitive recursive $+$, $*$, $\overline{sg}$, $|\cdot|$ are defined by $\widehat{R}_o$)

$$\left. \begin{array}{l} x =_o y \leftrightarrow |x-y| =_o 0 \\ x =_o 0 \wedge y =_o 0 \leftrightarrow x+y =_o 0 \\ x =_o 0 \vee y =_o 0 \leftrightarrow x*y =_o 0 \\ (x =_o 0 \to y =_o 0) \leftrightarrow \overline{sg}(x)*y =_o 0 \\ x \neq_o 0 \leftrightarrow \overline{sg}(x) =_o 0 \end{array} \right\} \tag{68}$$

- An axiom of quantifier-free-induction (below "$y < x$" is the usual primitive recursively definable strict order relation on natural numbers)

$$\mathtt{IA}'_0: \quad \forall f^1, x^o (f(0) =_o 0 \wedge \forall y < x (f(y) =_o 0 \to f(Sy) =_o 0) \to f(x) =_o 0) .$$

The $\mathtt{EIL}^\omega$-constants ν, I and E are immediately definable in $\mathtt{PRA}^\omega_\mathtt{i}$ from (68). Also the (here primitive-recursive) *closed* terms associated to *quantifier-free*-formulas A_0 (which here may also contain $\vee$) like in Proposition 3.21 are immediately provided in $\mathtt{PRA}^\omega_\mathtt{i}$ with $\mathtt{PRA}^\omega_\mathtt{i} \vdash t_{A_0}(\underline{a}) =_o 0 \leftrightarrow A_0(\underline{a})$. Because of this, $\mathtt{IA}'_0$ implies the following schema of quantifier-free-induction (below A_0 are quantifier-free-formulas which here may contain $\vee$):

$$\mathtt{IA}_0: \quad A_0(0) \wedge \forall x (A_0(x) \to A_0(Sx)) \to \forall x A_0(x) .$$

The $\mathtt{EIL}^\omega$-axiom $\mathtt{TND}_0: \ x =_o 0 \vee x \neq_o 0$ can be immediately derived from $\mathtt{IA}_0$. The $\mathtt{EIL}^\omega$-constant D can now easily be defined from $\widehat{R}$ in $\mathtt{PRA}^\omega_\mathtt{i}$.

The axioms $\mathtt{Ax}\widehat{R}$ and (68) are realizer-free except for the implication $x \cdot y =_o 0 \to (x =_o 0 \vee y =_o 0)$ whose functional interpretation is realized by $\Pi = \lambda x, y.\, x$. It follows that functional interpretation is immediately available for $\mathtt{PRA}^\omega_\mathtt{i}$ once realizing terms are provided for $\mathtt{IA}'_0$. Such terms of constant size can be built using Kleene recursors $\widehat{R}$ and are equivalent to

$$\lambda f, x.\, min_{\ y<x}\ t_{[\,f(0)=0 \wedge (\,f(y)=0 \to f(Sy)=0\,) \to f(x)=0\,]}(f,x,y) .$$

[39] For all ρ the recursor $\widehat{R}_\rho$ can be defined from $\widehat{R}_o$ using λ-abstraction and hence the $\mathtt{EIL}^\omega$ combinators in view of Definition 2.12. This property no longer holds for Gödel recursor R_ρ to be introduced in Section 5.3. See also Footnote 10 of [3].

Since $\geq$ and M are $\mathtt{PRA}^\omega_\mathtt{i}$-definable as well, the axioms added to $\mathtt{EIL}^\omega$ in Definition 4.11 become derivable in $\mathtt{PRA}^\omega_\mathtt{i}$. It follows that also monotone functional interpretation is available for $\mathtt{PRA}^\omega_\mathtt{i}$. In this case $\mathtt{IA}'_0$ is much simpler realized by projectors $\Pi = \lambda f, x.\, x$ (no recursors are needed).

Theorem 5.2 All the quantitative results proved above for $\mathtt{EIL}^\omega$ in Theorem 3.26 and $\mathtt{EIL}^\omega_\mathtt{M}$ in Theorem 4.18 carry on to $\mathtt{PRA}^\omega_\mathtt{i}$ in the obvious way. □

5.2 Extension to the analytical system $\mathtt{PRA}^\omega + \mathtt{AC}_0 + \mathtt{WKL}$.

The analogue of Theorem 5.2 for the classical system $\mathtt{PRA}^\omega + \mathtt{AC}_0$ holds as well. The system $\mathtt{PRA}^\omega + \mathtt{AC}_0$ allows to derive the schemas of Σ^0_1-induction and Δ^0_1-comprehension (see [30]) and therefore contains the system $\mathtt{RCA}_0$ known from reverse mathematics (see [49]).

Let us denote by $\mathtt{WKL}$ the binary König's lemma. This important[40] analytical principle simply asserts that every infinite binary tree has an infinite path. The second author has proved in [30] by means of a combination of functional interpretation and majorizability (a precursor of monotone functional interpretation) that $\mathtt{PRA}^\omega + \mathtt{AC}_0 + \mathtt{WKL}$ (which contains Friedman's system[41] $\mathtt{WKL}_0$ of [18, 49]) is Π^0_2-conservative over $\mathtt{PRA}^\omega_\mathtt{i}$. Moreover, a witnessing term can be provided. We give below a quantitative version of this result. We follow closely the proof in Section 7 of [3] which is a simplification of the more general method of [30].

Let $\mathtt{PRA}^\omega$ be formulated over $\mathtt{ECL}^\omega_\mathtt{M}$. We use the following convenient formulations of the binary König's lemma:

$$\mathtt{WKL} : \forall f\,[\,\forall k\,\neg \mathit{Bnd}(\mathit{BTr}(f), k) \;\rightarrow\; \exists b\,\forall k\,(\mathit{InSeg}(\mathit{Bin}(b), k) \in \mathit{BTr}(f))\,]$$

$$\mathtt{WKL}' : \forall f\,\exists b\,\forall k\,[\,\neg \mathit{Bnd}(\mathit{BTr}(f), k) \;\rightarrow\; \mathit{InSeg}(\mathit{Bin}(b), k) \in \mathit{BTr}(f)\,]$$

where (see Section 7 of [3] for full details)

- *Bin* and *BTr* are primitive recursive functionals which transform their argument to a binary function, respectively a binary tree;

[40] See [49] for a comprehensive discussion of the vast mathematical applicability of $\mathtt{WKL}$.

[41] Theorem I.10.3 of [49] gives a summary of important mathematical statements which are theorems of $\mathtt{WKL}_0$. We only mention here the Heine-Borel covering lemma, the maximum principle, the separable Hahn-Banach theorem and Brouwer's fixed point theorem.

- *InSeg* is a primitive recursive functional which produces the length k initial segment of the binary function $\mathit{Bin}(b)$;
- *Bnd* is a primitive recursive predicate which expresses that the given binary tree $BTr(f)$ has depth at most k.

Remark 5.3 Below A_0 is quantifier-free, $\underline{x}, \underline{y}$ are type o and $\{\underline{x}, \underline{y}\} = \mathrm{V_f}(A_0)$.

The following theorem expresses the fact that the WKL–elimination and term extraction procedure from WKL–based proofs as developed in [30] is feasible both w.r.t. the size of the extracted terms and the depth of the verifying WKL–free proof. Although the feasibility of WKL–elimination was first shown (for a fragment of second-order arithmetic) in [2], the forcing technique used there does not provide any term extraction.

Theorem 5.4 There exists $k \in \mathbb{N}$ constant and a functional-interpretation-based algorithm which does the following. Given as input a proof

$$\mathcal{P} : \mathtt{PRA}^\omega + \mathtt{AC}_0 \vdash_\partial \ \mathtt{WKL} \ \to \forall \underline{x} \, \exists \underline{y} \, A_0(\underline{x}, \underline{y})$$

it produces at output realizing terms $\underline{t}$ such that $Sz(\underline{t}) \leq k \cdot S_c(\mathcal{P})$ and

$$\mathtt{PRA}^\omega_{\mathtt{i}} \vdash_{k \cdot (ls_{\mathrm{o}} + \partial)} \forall \underline{x} \, A_0(\underline{x}, \underline{t}(\underline{x})) \ . \tag{69}$$

The time overhead of the algorithm is upper bounded by $k \cdot qs_{\mathrm{o}} \cdot ls_{\mathrm{o}} \cdot S_m(\mathcal{P})$.

Proof:

The first step is to transform $\mathcal{P} : \mathtt{PRA}^\omega + \mathtt{AC}_0 \vdash_\partial \ \mathtt{WKL} \ \to \forall \underline{x} \, \exists \underline{y} \, A_0(\underline{x}, \underline{y})$ to $\mathcal{P}^{\mathtt{N}} : \mathtt{PRA}^\omega_{\mathtt{i}} + \mathtt{AC}_0 + \mathtt{MK} \vdash_{k' \cdot (ld(A_0) + \partial)} \ \mathtt{WKL}' \ \to \forall \underline{x} \, \exists \underline{y} \, A_0(\underline{x}, \underline{y})$ such that all statements on $\mathcal{P}^{\mathtt{N}}$ in Proposition 4.4 hold. Here $\mathcal{P}^{\mathtt{N}}$ is obtained by a slight transformation within $\mathtt{PRA}^\omega_{\mathtt{i}} + \mathtt{MK}$ of the output from the N-translation algorithm carried on $\mathcal{P}$. There exist fixed proofs (hence with constant complexity) in $\mathtt{PRA}^\omega_{\mathtt{i}}$ of $\mathtt{WKL}' \to \mathtt{WKL}$ and $\mathtt{WKL} \to \mathtt{WKL}^{\mathtt{N}}$. See also Lemmas 7.3.1 and 7.3.3 of [3].

The second step is to transform $\mathcal{P}^{\mathtt{N}}$ to the proof in (69) via a technique based on functional interpretation and majorization. This technique is described in Lemmas 7.4.1 and 7.4.2 of [3] and is an adaptation of the more general technique of [30]. The elimination of $\mathtt{WKL}'$ is achieved by weakening $\mathtt{WKL}'$ to a formula which is provable in $\mathtt{PRA}^\omega_{\mathtt{i}}$. Since we are here interested also in the realizing term for $\exists \underline{y}$ and not only in the WKL–conservation, we use a (tuple-extended) variant of [3]–Lemma 7.4.1 where a realizer for $\exists y$ is provided as well. $\square$

Corollary 5.5 (quantitative WKL-conservation) There exists an algorithm which transforms proofs

$$\mathcal{P} : \mathtt{PRA}^\omega + \mathtt{AC}_0 \vdash_\partial \ \mathtt{WKL} \ \to \ \forall \underline{x} \, \exists \underline{y} \, A_0(\underline{x}, \underline{y})$$

into proofs

$$\mathcal{P}' : \mathtt{PRA}^\omega_{\mathtt{i}} \vdash_{k\cdot(ls_{\mathrm{o}}+\partial)} \forall \underline{x} \, \exists \underline{y} \, A_0(\underline{x}, \underline{y})$$

Remark 5.6 We could alternatively use a monotone functional interpretation – version of [3]–Lemma 7.4.1 in the lines of Theorem 4.20. Then we would first obtain a majorizing tuple $\underline{t^*}$ for $\exists \underline{y}$ and we could produce a realizer by bounded search (up to $\underline{t^*}(\underline{x})$) along the predicate $t_{A_0}(\underline{x}, \underline{y}) = 0$. In such a case Theorem 5.4 would hold with (69) replaced by

$$\mathtt{PRA}^\omega_{\mathtt{i}} \vdash_{k\cdot(ld(A_0)+qs_{\mathrm{o}}+\partial)} \forall \underline{x} \, A_0(\underline{x}, \underline{t}(\underline{x})) \ .$$

In many cases A_0 is monotone in $\underline{y}$ and therefore bounded search is actually not needed[42]. In such a case we would obtain terms $\underline{t}$ with $Sz(\underline{t}) \leq k \cdot S_m(\mathcal{P})$, time overhead at most $k \cdot fid_{\mathrm{o}} \cdot qs_{\mathrm{o}} \cdot S_m(\mathcal{P})$ and

$$\mathtt{PRA}^\omega_{\mathtt{i}} \vdash_{k\cdot(qs_{\mathrm{o}}+\partial)} \forall \underline{x} \, A_0(\underline{x}, \underline{t}(\underline{x})) \tag{70}$$

hence a full better performance than the algorithm of Theorem 5.4.

Remark 5.7 There are three ways to produce a variant of Theorem 5.4 where the input proof is $\mathcal{P} : \mathtt{PRA}^\omega + \mathtt{AC}_0 + \mathtt{WKL} \vdash_\partial \ \forall \underline{x} \, \exists \underline{y} \, A_0(\underline{x}, \underline{y})$. One way to overcome the failure of the deduction theorem for weakly extensional $\mathtt{PRA}^\omega$ is via the elimination-of-extensionality procedure from [42]. This applies only when $\mathcal{P}$ contains just[43] variables of type 0 or 1, but in fact this is the case in most applications. We conjecture that the aforementioned procedure is feasible and hence the overall term extraction and WKL-conservation is still a feasible process. However if we are mainly interested in the term extraction rather than the WKL-conservation we can state a monotone-functional-interpretation-based variant of Theorem 5.4 with the verifying proof in $\mathtt{PRA}^\omega_{\mathtt{i}} + \widetilde{\mathtt{WKL}}$ and of the same depth as (70), where

[42] See also the remarks following Theorem 4.20.

[43] Under this type restriction we can allow the use of (full) extensionality axiom EA, see also Remark 2.6. Hence in this setting we work with (fully) extensional $\mathtt{PRA}^\omega$ which features the deduction theorem.

$\widetilde{\mathtt{WKL}}:\ \exists B\,\forall f\,\forall k\,[\,\neg \mathit{Bnd}(\mathit{BTr}(f),k)\ \rightarrow\ \mathit{InSeg}(\mathit{Bin}(B(f)),k)\in \mathit{BTr}(f)\,]$

is a strengthening of $\mathtt{WKL}'$. If we are satisfied with a partial $\mathtt{WKL}$-conservation then we can use the fact that premises of $\mathtt{ER}_0$ are realizer-free and hence any $\mathtt{WKL}$ instance used in the proof of such a $\mathtt{ER}_0$-premise gets discarded in the preprocessing phase of the (monotone) functional-interpretation-algorithm. We can thus consider that the input proof is in $\mathtt{PRA}^\omega+\mathtt{AC}_0\oplus\mathtt{WKL}$ (see [30], p. 1246 for the meaning of $\oplus$ in this context). For this system the deduction theorem holds w.r.t. $\oplus$ and we obtain (69) with $\mathtt{PRA}^\omega_{\mathtt{i}}$ extended with the N-translations of conclusions of those $\mathtt{ER}_0$ instances in $\mathcal{P}$ whose sub-proof-trees use $\mathtt{WKL}$. See also Remark 3.32.

Remark 5.8 Even though the term extraction procedure of Theorem 5.4 is extremely feasible, the normalization of the extracted terms into ordinary primitive recursive functions and the verification in (plain) primitive recursive arithmetic would however trigger a non-elementary recursive complexity.

5.3 The case of Peano Arithmetic $\mathtt{PA}^\omega$ and $\mathtt{PA}^\omega+\mathtt{AC}_0+\mathtt{WKL}$

Already Gödel showed that the functional interpretation of full induction can be realized by his impredicative recursors $\underline{R}$ for (simultaneous[44]) primitive recursion in finite types, where (below $i\in\overline{1,|\underline{\sigma}|}$ with $|\underline{\sigma}|$ the length of $\underline{\sigma}$)

$$\mathtt{Ax}R:\quad \begin{cases} R^i_{\underline{\sigma}}(\underline{x},\underline{y},0) &=_{\sigma_i}\ x_i\\ R^i_{\underline{\sigma}}(\underline{x},\underline{y},Sz) &=_{\sigma_i}\ y_i(R^1_{\underline{\sigma}}(\underline{x},\underline{y},z),\ldots,R^{|\underline{\sigma}|}_{\underline{\sigma}}(\underline{x},\underline{y},z),z)\end{cases}.$$

Gödel's objective was to reduce the consistency of Peano arithmetic $\mathtt{PA}$ to that of a quantifier-free calculus (called $\mathbf{T}$) based on these R. In order to achieve this he had to give a verifying proof for the functional interpretation of induction which used only quantifier-free induction. For the applied purpose of program extraction this is not required. We may use full induction in the verifying proof as well. This simplifies matters substantially as was already observed in [53](3.5.5.(iii)). Things are particularly simple if induction is formulated as a rule

$$\mathtt{IR}:\quad \frac{A(0)\,,\ A(z)\rightarrow A(Sz)}{A(z)}$$

[44]Simultaneous primitive recursion even in higher types can be reduced to ordinary primitive recursion in higher types. This is particularly simple in the presence of $\mathtt{ER}_0$, see [53].

which nevertheless allows to derive the axiom schema of induction

$\mathtt{IA}: \quad A(0) \wedge \forall z\,(\,A(z) \to A(Sz)\,) \to \forall z A(z)\ .$

The treatment of IR under functional interpretation is fairly similar to that of modus ponens MP. Recursors R have to be used in addition to Σ. Given that $\underline{t_1}$ Dr $A(z,\underline{a})$ and $(\underline{t_2},\underline{t_3})$ Dr $(A(z,\underline{a}) \to A(Sz,\underline{a}))$ one can prove[45] in a constant number of steps that $\underline{t_4}$ Dr $\forall z A(z,\underline{a})$. Here $t_4 \equiv \Sigma\, R\, \underline{t_1}\,(P\, t_3^1)\ldots(P\, t_3^{|t_3|})$. The monotone functional interpretation of IR is much the same as the usual one if we use recursors R^* which can easily be defined from R by a minor modification (see also [24] for similar R^+ recursors). Negative translation applies to IR just as it did for MP. Let Peano arithmetic in all finite types $\mathtt{PA}^\omega$ be formulated over $\mathtt{ECL}^\omega$ plus IR and recursors R. The remarks above imply that we can state the following consequence of (Corollary 4.10) of Theorem 4.8.

Theorem 5.9 There exists $k \in \mathbb{N}$ constant and a functional-interpretation-based algorithm which does the following. Given as input a proof $\mathcal{P} : \mathtt{PA}^\omega + \mathtt{AC}_0 \vdash_\partial \forall \underline{x}\, \exists \underline{y}\, A_0(\underline{x},\underline{y})$ it produces a realizing tuple $\underline{t}$ such that $\mathtt{PA}^\omega_{\mathtt{i}} \vdash_{k\cdot(ls_0+\partial)} \forall \underline{x}\, A_0(\underline{x},\underline{t}(\underline{x}))$ and $Sz(\underline{t}) \le k \cdot S_c(\mathcal{P})$. The time overhead of the algorithm is upper bounded by $k \cdot qs_0 \cdot ls_0 \cdot S_m(\mathcal{P})$. □

In contrast to IR, the treatment of IA under usual functional interpretation results in complexity issues similar to those of CT$\wedge$ in Proposition 3.22. This is hinted by the fact that the derivation of IA from IR apparently needs contraction $A \to A \wedge A$. Let A be the induction formula; in order to realize IA we need bounded search along the predicate $t_{A_\mathtt{D}}(\underline{x},y) =_o 0$. Here $t_{A_\mathtt{D}}$ is a characteristic term for $A_\mathtt{D}$, see [45]. The monotone functional interpretation of IA avoids this altogether (like before in the case of $A \to A \wedge A$). Now only a majorizing term for $\mu y \le z\,[\,t_{A_\mathtt{D}}(\underline{x},y) =_o 0\,]$ needs to be constructed and we can simply use $t^* :\equiv \Pi = \lambda \underline{x}, z.\, z$. Let $\mathtt{PA}^\omega$ be formulated over $\mathtt{ECL}^\omega_\mathtt{M}$ plus IA and the recursors R. The remarks above imply that we can state the following result.

Theorem 5.10 All the quantitative results of Theorem 5.4 and the subsequent considerations carry on to the corresponding $\mathtt{PA}^\omega$-based systems in the obvious way. □

[45] Obviously using IR.

Acknowledgements: The authors are grateful to Paulo Oliva, Philipp Gerhardy and Daniele Varacca for their comments and suggestions on an earlier draft of this work. We also thank Prof. Helmut Schwichtenberg for useful comments.

References

[1] W. Alexi. Extraction and verification of programs by analysis of formal proofs. *Theoretical Computer Science*, 61:225–258, 1988.

[2] J. Avigad. Formalizing forcing arguments in subsystems of second-order arithmetic. *Annals of Pure and Applied Logic*, 82:165–191, 1996.

[3] J. Avigad and S. Feferman. Gödel's functional ('Dialectica') interpretation. In [9], pages 337–405.

[4] J. Barwise, editor. *Handbook of Mathematical Logic*, volume 90 of *Studies in Logic and the Foundations of Mathematics*. North-Holland, Amsterdam, New York, Oxford, 1977.

[5] S. Bellantoni and H. Schwichtenberg. Feasible computation with higher types. Marktoberdorf 2001 volume.

[6] U. Berger, W. Buchholz, and H. Schwichtenberg. Refined program extraction from classical proofs. *Annals of Pure and Applied Logic*, 114:3–25, 2002.

[7] J. Blanck and al., editors. *Proceedings Fourth Workshop on Computability and Complexity in Analysis (CCA 2000)*, volume 2064 of *Springer LNCS*. Springer, 2001.

[8] W. Burr. Functional interpretation of Aczel's constructive set theory. *Ann. Pure Applied Logic*, 104:31–75, 2000.

[9] S.R. Buss, editor. *Handbook of Proof Theory*, volume 137 of *Studies in Logic and the Foundations of Mathematics*. Elsevier, 1998.

[10] S. Cook and A. Urquhart. Functional interpretations of feasibly constructive arithmetic. *Ann. Pure Applied Logic*, 63:103–200, 1993.

[11] T. Coquand and M. Hofmann. A new method for establishing conservativity of classical systems over their intuitionistic version. *Math. Structures Comput. Sci.*, 9(4):323–333, 1999.

[12] J.C.E. Dekker, editor. *Recursive function theory, Symposia in Pure Mathematics*, volume 5, 1962.

[13] J. Diller and W. Nahm. Eine Variante zur Dialectica Interpretation der Heyting Arithmetik endlicher Typen. *Archive für Mathematische Logik und Grundlagenforschung*, 16:49–66, 1974.

[14] A.G. Dragalin. New kinds of realisability and the Markov rule. *Dokl. Akad. Nauk. SSSR*, 251:534–537, 1980. Russian, English Translation [15].

[15] A.G. Dragalin. New kinds of realisability and the Markov rule. *Soviet Math. Dokl.*, 21:461–464, 1980.

[16] S. Feferman. Theories of finite type related to mathematical practice. In [4], pages 913–972.

[17] H. Friedman. Classical and intuitionistically provably recursive functions. In [43], pages 21–27.

[18] H. Friedman. Systems of second order arithmetic with restricted induction, I, II (abstracts). *Journal of Symbolic Logic*, 41:557–559, 1976.

[19] J.-Y. Girard. *Interprétation fonctionnelle et élimination des coupures dans l'arithmétique d'ordre supérior.* PhD thesis, Université de Paris VII, 1972.

[20] K. Gödel. Zur intuitionistischen Arithmetik und Zahlentheorie. *Ergebnisse eines Mathematischen Kolloquiums*, 4:34–38, 1933.

[21] K. Gödel. Über eine bisher noch nicht benützte Erweiterung des finiten Standpunktes. *Dialectica*, 12:280–287, 1958.

[22] S. Hayashi and H. Nakano. *PX: A Computational Logic.* MIT Press, 1988.

[23] W. Hodges, M. Hyland, C. Steinhorn, and J. Truss, editors. *Logic: from Foundations to Applications. European Logic Colloquium, Keele, 1993.* Oxford University Press, 1996.

[24] W.A. Howard. Hereditarily majorizable functionals of finite type. In [53], pages 454–461.

[25] K.F. Joergensen. Finite type arithmetic. Master's thesis, University of Roskilde, Departments of Mathematics and Philosophy, 2001.

[26] U. Kohlenbach. Analysing proofs in analysis. In [23], pages 225–260.

[27] U. Kohlenbach. Lecture Course: Proof Interpretations and the Computational Content of Proofs. Updated version at http://www.brics.dk/~kohlenb.

[28] U. Kohlenbach. On the computational content of the Krasnoselski and Ishikawa fixed point theorems. In [7], pages 119–145.

[29] U. Kohlenbach. Uniform asymptotic regularity for Mann iterates. To appear in J. Math. Anal. Appl.

[30] U. Kohlenbach. Effective bounds from ineffective proofs in analysis: an application of functional interpretation and majorization. *J. Symbolic Logic*, 57:1239–1273, 1992.

[31] U. Kohlenbach. Effective moduli from ineffective uniqueness proofs. An unwinding of de La Vallée Poussin's proof for Chebycheff approximation. *Ann. Pure Appl. Logic*, 64:27–94, 1993.

[32] U. Kohlenbach. New effective moduli of uniqueness and uniform a–priori estimates for constants of strong unicity by logical analysis of known proofs in best approximation theory. *Numer. Funct. Anal. and Optimiz.*, 14:581–606, 1993.

[33] U. Kohlenbach. Mathematically strong subsystems of analysis with low rate of growth of provably recursive functionals. *Arch. Math. Logic*, 36:31–71, 1996.

[34] U. Kohlenbach. On the no-counterexample interpretation. *Journal of Symbolic Logic*, 64:1491–1511, 1999.

[35] U. Kohlenbach. A quantitative version of a theorem due to Borwein-Reich-Shafrir. *Numer. Funct. Anal. and Optimiz.*, 22:641–656, 2001.

[36] U. Kohlenbach and L. Leuştean. Mann iterates of directionally nonexpansive mappings in hyperbolic spaces. To appear in Abstr. Appl. Anal.

[37] U. Kohlenbach and P. Oliva. Proof mining: a systematic way of analysing proofs in Mathematics. To appear in Proc. Steklov Inst. Math.

[38] U. Kohlenbach and P. Oliva. Proof mining in L_1-approximation. To appear in Ann. Pure Appl. Logic.

[39] G. Kreisel. On the interpretation of non-finitist proofs, part I. *Journal of Symbolic Logic*, 16:241–267, 1951.

[40] G. Kreisel. On the interpretation of non-finitist proofs, part II: Interpretation of number theory. *Journal of Symbolic Logic*, 17:43–58, 1952.

[41] G. Kreisel. On weak completeness of intuitionistic predicate logic. *J. Symbolic Logic*, 27:139–158, 1962.

[42] H. Luckhardt. Extensional Gödel Functional Interpretation. *Springer LNM*, 306, 1973.

[43] G.H. Müller and D.S. Scott, editors. *Higher Set Theory*, volume 669 of *Lecture Notes in Mathematics*. Springer, 1978.

[44] C. Murthy. *Extracting Constructive Content from Classical Proofs*. PhD thesis, Cornell University, 1990.

[45] C. Parsons. On n-quantifier induction. *J. Symbolic Logic*, 37:466–482, 1972.

[46] P. Rath. *Eine verallgemeinerte Funktionalinterpretation der Heyting Arithmetik endlicher Typen*. PhD thesis, Universität Münster, 1978.

[47] M. Schönfinkel. Über die Bausteine der mathematischen Logik. *Math. Annalen*, 92:305–316, 1924.

[48] H. Schwichtenberg. An arithmetic for polynomial-time computation. Submitted. Available at http://www.mathematik.uni-muenchen.de/~schwicht.

[49] S.G. Simpson. *Subsystems of Second Order Arithmetic*. Perspectives in Mathematical Logic. Springer-Verlag, 1999.

[50] C. Spector. Provably recursive functionals of analysis: a consistency proof of analysis by an extension of principles formulated in current intuitionistic mathematics. In [12], pages 1–27.

[51] M. Stein. *Interpretation der Heyting-Arithmetik endlicher Typen*. PhD thesis, Universität Münster, 1976.

[52] A.S. Troelstra. Realisability. In [9], pages 407–473.

[53] A.S. Troelstra, editor. *Metamathematical investigation of intuitionistic arithmetic and analysis*, volume 344 of *Lecture Notes in Mathematics*. Springer-Verlag, Berlin - Heidelberg - New York, 1973.

A Appendix

Below we complete the proof of Proposition 3.20. We first recall its statement.

Proposition A.1 There exists $k \in \mathbb{N}$ constant such that for any instance A of $\mathtt{CT}\vee$, $\mathtt{WK}\vee$, $\mathtt{WK}\wedge$, $\mathtt{PM}\vee$, $\mathtt{PM}\wedge$, $\mathtt{SYL}$, $\mathtt{EPN}$, $\mathtt{EFQ}$, $\mathtt{TND}_0$, $\mathtt{MK}$, $\mathtt{IP}_\forall$, $\mathtt{AC}$, there exists a realizing tuple for $A^{\mathtt{D}}$ such that

$$\begin{array}{rcll} d(\underline{t}) & \leq & k & (71)\\ S(\underline{t}) & \leq & k & (72)\\ mdg(\underline{t}) & \leq & k + vdg(A) + id(A) & (73)\\ mar(\underline{t}) & \leq & k + var(A) + qs(A)\,(id(A) - k_0 + 2) & (74)\\ & \vdash_k & \{\!|\, \underline{t}\,,\, A \,|\!\} & (75) \end{array}$$

Proof: Only $\mathtt{SYL}$ was treated in the proof of Proposition 3.20. Below we treat the remaining axioms, where $\{\underline{a}\} \equiv \mathrm{V_f}(\textit{the corresp. axiom instance})$.

$\mathtt{CT}\vee$: By definition,

$$(A \vee A \to A)^{\mathtt{D}} \equiv \exists \underline{Y}, \underline{Y'}, \underline{X''}\ \forall z, \underline{x}, \underline{x'}, \underline{y''}$$
$$\left(\begin{array}{c} \left(\begin{array}{c} z = 0 \to A_{\mathtt{D}}(\underline{x}; \underline{Y}(z, \underline{x}, \underline{x'}, \underline{y''})) \\ \wedge \\ z \neq 0 \to A_{\mathtt{D}}(\underline{x'}; \underline{Y'}(z, \underline{x}, \underline{x'}, \underline{y''})) \end{array}\right) \\ \longrightarrow \\ A_{\mathtt{D}}(\underline{X''}(z, \underline{x}, \underline{x'}); \underline{y''}) \end{array}\right)$$

and we can take $\left\{\begin{array}{rcl} t_Y & :\equiv & t_{Y'} :\equiv \Pi = \lambda \underline{a}, z, \underline{x}, \underline{x'}, \underline{y''}.\, y'' \\ t_{X''} & :\equiv & \Pi\, D = \lambda \underline{a}.\, D \end{array}\right.$

$\mathtt{WK}\vee$: By definition,

$$(A \to A \vee B)^{\mathtt{D}} \equiv \exists \underline{Y}, Z, \underline{X'}, \underline{U}\ \forall \underline{x}, \underline{y'}, \underline{v}$$
$$\left(\begin{array}{c} A_{\mathtt{D}}(\underline{x}; \underline{Y}(\underline{x}, \underline{y'}, \underline{v})) \\ \longrightarrow \\ \left(\begin{array}{c} Z(\underline{x}) = 0 \to A_{\mathtt{D}}(\underline{X'}(\underline{x}); \underline{y'}) \\ \wedge \\ Z(\underline{x}) \neq 0 \to B_{\mathtt{D}}(\underline{U}(\underline{x}); \underline{v}) \end{array}\right) \end{array}\right)$$

$$\text{and we can take } \left\{ \begin{array}{rcl} t_Y & :\equiv & \Pi = \lambda \underline{a}, \underline{x}, \underline{y'}, \underline{v}.\, y' \\ t_Z & :\equiv & O = \lambda \underline{a}, \underline{x}.\, 0 \\ t_{X'} & :\equiv & \Pi = \lambda \underline{a}, \underline{x}.\, x \\ t_U & :\equiv & O \end{array} \right.$$

$\mathtt{WK}\wedge$: By definition,

$$(A \wedge B \to A)^{\mathtt{D}} \equiv \exists \underline{Y}, \underline{V}, \underline{X'}\, \forall \underline{x}, \underline{u}, \underline{y'}$$
$$\left(\begin{array}{c} A_{\mathtt{D}}(\underline{x}; \underline{Y}(\underline{x}, \underline{u}, \underline{y'})) \wedge B_{\mathtt{D}}(\underline{u}; \underline{V}(\underline{x}, \underline{u}, \underline{y'})) \\ \longrightarrow \\ A_{\mathtt{D}}(\underline{X'}(\underline{x}, \underline{u}); \underline{y'}) \end{array} \right)$$

$$\text{and we can take } \left\{ \begin{array}{rcl} t_Y & :\equiv & \Pi = \lambda \underline{a}, \underline{x}, \underline{u}, \underline{y'}.\, y' \\ t_V & :\equiv & O \\ t_{X'} & :\equiv & \Pi = \lambda \underline{a}, \underline{x}, \underline{u}.\, x \end{array} \right.$$

$\mathtt{PM}\vee$: By definition,

$$(A \vee B \to B \vee A)^{\mathtt{D}} \equiv \exists \underline{Y}, \underline{V}, Z', \underline{U'}, \underline{X'}\, \forall z, \underline{x}, \underline{u}, \underline{v'}, \underline{y'}$$
$$\left(\begin{array}{c} \left(\begin{array}{c} z = 0 \to A_{\mathtt{D}}(\underline{x}; \underline{Y}(z, \underline{x}, \underline{u}, \underline{v'}, \underline{y'})) \\ \wedge \\ z \neq 0 \to B_{\mathtt{D}}(\underline{u}; \underline{V}(z, \underline{x}, \underline{u}, \underline{v'}, \underline{y'})) \end{array} \right) \\ \longrightarrow \\ \left(\begin{array}{c} \underline{Z'}(z, \underline{x}, \underline{u}) = 0 \to B_{\mathtt{D}}(\underline{U'}(z, \underline{x}, \underline{u}); \underline{v'}) \\ \wedge \\ \underline{Z'}(z, \underline{x}, \underline{u}) \neq 0 \to A_{\mathtt{D}}(\underline{X'}(z, \underline{x}, \underline{u}); \underline{y'}) \end{array} \right) \end{array} \right)$$

$$\text{and we can take } \left\{ \begin{array}{rcl} t_Y & :\equiv & \Pi = \lambda \underline{a}, z, \underline{x}, \underline{u}, \underline{v'}, \underline{y'}.\, y' \\ t_V & :\equiv & \Pi = \lambda \underline{a}, z, \underline{x}, \underline{u}, \underline{v'}, \underline{y'}.\, v' \\ t_{Z'} & :\equiv & \Sigma\, (\Sigma\, (\Pi\, I)\, \Pi)\, (\Pi\, 1) = \lambda \underline{a}, z, \underline{x}, \underline{u}.\, (I\, z\, 1) \\ t_{U'} & :\equiv & \Pi = \lambda \underline{a}, z, \underline{x}, \underline{u}.\, u \\ t_{X'} & :\equiv & \Pi = \lambda \underline{a}, z, \underline{x}, \underline{u}.\, x \end{array} \right.$$

$\mathtt{PM}\wedge$: By definition,

$$(A \wedge B \to B \wedge A)^{\mathtt{D}} \equiv \exists \underline{Y}, \underline{V}, \underline{U'}, \underline{X'}\, \forall \underline{x}, \underline{u}, \underline{y'}, \underline{v'}$$
$$\left(\begin{array}{c} A_{\mathtt{D}}(\underline{x}; \underline{Y}(\underline{x}, \underline{u}, \underline{y'}, \underline{v'})) \wedge B_{\mathtt{D}}(\underline{u}; \underline{V}(\underline{x}, \underline{u}, \underline{y'}, \underline{v'})) \\ \to \\ B_{\mathtt{D}}(\underline{U'}(\underline{x}, \underline{u}); \underline{v'}) \wedge A_{\mathtt{D}}(\underline{X'}(\underline{x}, \underline{u}); \underline{y'}) \end{array} \right)$$

and we can take $\left\{ \begin{array}{lcl} t_Y & :\equiv & \Pi = \lambda \underline{a}, \underline{x}, \underline{u}, \underline{y'}, \underline{v'}.\, y' \\ t_V & :\equiv & \Pi = \lambda \underline{a}, \underline{x}, \underline{u}, \underline{y'}, \underline{v'}.\, v' \\ t_{U'} & :\equiv & \Pi = \lambda \underline{a}, \underline{x}, \underline{u}.\, u \\ t_{X'} & :\equiv & \Pi = \lambda \underline{a}, \underline{x}, \underline{u}.\, x \end{array} \right.$

$\mathtt{EFQ}$: By definition,

$$(1 = 0 \to A)^{\mathtt{D}} \equiv \exists \underline{x}\, \forall \underline{y}\, (1 = 0 \to A_{\mathtt{D}}(\underline{x}; \underline{y}))$$

and we can take $t_x :\equiv O$. $\square$

$\mathtt{EPN}$: By definition,

$$\begin{array}{rcl} (A \to B)^{\mathtt{D}} & \equiv & \exists \underline{Y}, \underline{U}\, \forall \underline{x}, \underline{v}\, (A_{\mathtt{D}}(\underline{x}; \underline{Y}(\underline{x}, \underline{v})) \to B_{\mathtt{D}}(\underline{U}(\underline{x}); \underline{v})) \\ (C \vee A)^{\mathtt{D}} & \equiv & \exists z, \underline{g}, \underline{x}\, \forall \underline{h}, \underline{y}\, ((z = 0 \to C_{\mathtt{D}}(\underline{g}; \underline{h})) \wedge (z \neq 0 \to A_{\mathtt{D}}(\underline{x}; \underline{y}))) \\ (C \vee B)^{\mathtt{D}} & \equiv & \exists z', \underline{g'}, \underline{u}\, \forall \underline{h'}, \underline{v}\, ((z' = 0 \to C_{\mathtt{D}}(\underline{g'}; \underline{h'})) \wedge (z' \neq 0 \to B_{\mathtt{D}}(\underline{u}; \underline{v}))) \end{array}$$

hence

$$(C \vee A \to C \vee B)^{\mathtt{D}} \equiv \exists \underline{H}, \underline{Y}, Z', \underline{G'}, \underline{U'}\, \forall z, \underline{g}, \underline{x}, \underline{h'}, \underline{v}$$
$$\left(\begin{array}{c} \left(\begin{array}{c} z = 0 \to C_{\mathtt{D}}(\underline{g}; \underline{H}(z, \underline{g}, \underline{x}, \underline{h'}, \underline{v})) \\ \wedge \\ z \neq 0 \to A_{\mathtt{D}}(\underline{x}; \underline{Y}(z, \underline{g}, \underline{x}, \underline{h'}, \underline{v})) \end{array} \right) \\ \longrightarrow \\ \left(\begin{array}{c} Z'(z, \underline{g}, \underline{x}) = 0 \to C_{\mathtt{D}}(\underline{G'}(z, \underline{g}, \underline{x}); h') \\ \wedge \\ Z'(z, \underline{g}, \underline{x}) \neq 0 \to B_{\mathtt{D}}(\underline{U'}(z, \underline{g}, \underline{x}); \underline{v}) \end{array} \right) \end{array} \right)$$

and further

$$((A \to B) \to (C \vee A \to C \vee B))^{\mathtt{D}} \equiv \exists \underline{X}, \underline{V}, \underline{H}, \underline{Y}, Z', \underline{G'}, \underline{U'}\ \forall \underline{y}, \underline{u}, z, \underline{g}, \underline{x}, \underline{h'}, \underline{v}$$

$$\left(\begin{array}{c} \left(\begin{array}{c} A_{\mathtt{D}}(\underline{X}(\underline{y}, \underline{u}, z, \underline{g}, \underline{x}, \underline{h'}, \underline{v}); \underline{y}(\underline{X}(\underline{y}, \underline{u}, z, \underline{g}, \underline{x}, \underline{h'}, \underline{v}), \underline{V}(\underline{y}, \underline{u}, z, \underline{g}, \underline{x}, \underline{h'}, \underline{v}))) \\ \to \\ B_{\mathtt{D}}(\underline{u}(\underline{X}(\underline{y}, \underline{u}, z, \underline{g}, \underline{x}, \underline{h'}, \underline{v})); \underline{V}(\underline{y}, \underline{u}, z, \underline{g}, \underline{x}, \underline{h'}, \underline{v})) \end{array} \right) \\ \longrightarrow \\ \left(\begin{array}{c} z = 0 \to C_{\mathtt{D}}(\underline{g}; \underline{H}(\underline{y}, \underline{u})(z, \underline{g}, \underline{x}, \underline{h'}, \underline{v})) \\ \wedge \\ z \neq 0 \to A_{\mathtt{D}}(\underline{x}; \underline{Y}(\underline{y}, \underline{u})(z, \underline{g}, \underline{x}, \underline{h'}, \underline{v})) \end{array} \right) \\ \longrightarrow \\ \left(\begin{array}{c} Z'(\underline{y}, \underline{u})(z, \underline{g}, \underline{x}) = 0 \to C_{\mathtt{D}}(\underline{G'}(\underline{y}, \underline{u})(z, \underline{g}, \underline{x}); \underline{h'}) \\ \wedge \\ Z'(\underline{y}, \underline{u})(z, \underline{g}, \underline{x}) \neq 0 \to B_{\mathtt{D}}(\underline{U'}(\underline{y}, \underline{u})(z, \underline{g}, \underline{x}); \underline{v}) \end{array} \right) \end{array} \right)$$

and we can take

$$\left\{ \begin{array}{lcl} t_X & :\equiv & \Pi = \lambda \underline{a}, \underline{y}, \underline{u}, z, \underline{g}, \underline{x}, \underline{h'}, \underline{v}.\, x \\ t_V & :\equiv & \Pi = \lambda \underline{a}, \underline{y}, \underline{u}, z, \underline{g}, \underline{x}, \underline{h'}, \underline{v}.\, v \\ t_H & :\equiv & \Pi = \lambda \underline{a}, \underline{y}, \underline{u}, z, \underline{g}, \underline{x}, \underline{h'}, \underline{v}.\, h' \\ t_{Z'} & :\equiv & \Pi = \lambda \underline{a}, \underline{y}, \underline{u}, z, \underline{g}, \underline{x}.\, z \\ t_Y & :\equiv & P\,\Sigma = \lambda \underline{a}, \underline{y}, \underline{u}, z, \underline{g}, \underline{x}, \underline{h'}.\, y(\underline{x}) \\ t_{G'} & :\equiv & \Pi = \lambda \underline{a}, \underline{y}, \underline{u}, z, \underline{g}, \underline{x}, \underline{h'}, \underline{v}.\, g \\ t_{U'} & :\equiv & \Pi = \lambda \underline{a}, \underline{y}, \underline{u}, z, \underline{g}.\, u \end{array} \right.$$

$\mathtt{TND}_0$: By definition,

$$[\, x = 0 \ \vee\ \neg(x = 0)\,]^{\mathtt{D}} \equiv \exists z\, [\, (z = 0 \to x = 0)\ \wedge\ (Iz1 = 0 \to \neg(x = 0))\,]$$

and we can take $t_z :\equiv \lambda x.\, x$.

The remaining axioms are of shape $A \to B$ such that $A^{\mathtt{D}} \equiv B^{\mathtt{D}}$ and therefore are immediately seen to be realized with projectors Π. For the reader's convenience we nevertheless give below the full details.

MK: We have $[\, \neg\neg\, \exists \underline{x}\, A_0(\underline{x})\,]^{\mathtt{D}} \equiv \exists \underline{x}\, \neg\neg\, A_0(\underline{x})$ hence

$$[\, \neg\neg\, \exists \underline{x}\, A_0(\underline{x})\ \to\ \exists \underline{x}\, \neg\neg\, A_0(\underline{x})\,]^{\mathtt{D}} \equiv \exists \underline{X}\, \forall \underline{x}\, [\, \neg\neg\, A_0(\underline{x})\ \to\ \neg\neg\, A_0(\underline{X}(\underline{x}))\,]$$

and we can take $t_X :\equiv \Pi = \lambda \underline{a}, \underline{x}.\, x$.

$\mathtt{IP}_\forall$: We have

$$\begin{array}{rcl} [\,\forall \underline{x}\, A_0(\underline{x}) \,\to\, \exists \underline{y}\, B(\underline{y})\,]^{\mathtt{D}} & \equiv & \exists \underline{X}, \underline{y}, \underline{u}\, \forall \underline{v}\, [\, A_0(\underline{X}(\underline{v})) \,\to\, B_{\mathtt{D}}(\underline{u}; \underline{v}; \underline{y})\,] \\ (\,\exists \underline{y}\, [\,\forall \underline{x}\, A_0(\underline{x}) \,\to\, B(\underline{y})\,]\,)^{\mathtt{D}} & \equiv & \exists \underline{y}, \underline{X}, \underline{u}\, \forall \underline{v}\, [\, A_0(\underline{X}(\underline{v})) \,\to\, B_{\mathtt{D}}(\underline{u}; \underline{v}; \underline{y})\,] \end{array}$$

hence $(\,[\,\forall \underline{x}\, A_0(\underline{x}) \,\to\, \exists \underline{y}\, B(\underline{y})\,] \,\to\, \exists \underline{y}\, [\,\forall \underline{x}\, A_0(\underline{x}) \,\to\, B(\underline{y})\,]\,)^{\mathtt{D}}$ is

$$\exists V, Y, X, U\, \forall x, y, u, v$$
$$\left(\begin{array}{c} A_0(x(V(x,y,u,v))) \,\to\, B_{\mathtt{D}}(\underline{u}; V(x,y,u,v); y) \\ \longrightarrow \\ A_0(X(x,y,u,v)) \,\to\, B_{\mathtt{D}}(U(x,y,u); v; Y(x,y,u)) \end{array} \right)$$

and we can take $\left\{ \begin{array}{rcl} t_V & :\equiv & \Pi = \lambda \underline{a}, \underline{x}, \underline{y}, \underline{u}, \underline{v}.\, v \\ t_Y & :\equiv & \Pi = \lambda \underline{a}, \underline{x}, \underline{y}, \underline{u}.\, y \\ t_X & :\equiv & \Pi = \lambda \underline{a}, \underline{x}, \underline{y}, \underline{u}.\, x \\ t_U & :\equiv & \Pi = \lambda \underline{a}, \underline{x}, \underline{y}, \underline{u}.\, u \end{array} \right.$

$\mathtt{AC}$: We have

$$[\,\forall x\, \exists y\, B(\underline{x}, \underline{y})\,]^{\mathtt{D}} \equiv \exists Y, \underline{U}\, \forall x, v\, B_{\mathtt{D}}(\underline{U}(x); v; x, Y(x)) \equiv [\,\exists Y\, \forall x\, B(x, Y(x))\,]^{\mathtt{D}}$$

hence $[\,\forall \underline{x}\, \exists \underline{y}\, B(\underline{x}, \underline{y}) \,\to\, \exists \underline{Y}\, \forall \underline{x}\, B(\underline{x}, \underline{Y}(\underline{x}))\,]^{\mathtt{D}}$ is

$$\exists X, V, Y, U\, \forall y, u, x, v$$
$$\left(\begin{array}{c} B_{\mathtt{D}}(u(X(y,u,x,v)); V(y,u,x,v); X(y,u,x,v), y(X(y,u,x,v))) \\ \longrightarrow \\ B_{\mathtt{D}}(U(y,u,x); v; x, Y(y,u,x)) \end{array} \right)$$

and we can take $\left\{ \begin{array}{rcl} t_X & :\equiv & \Pi = \lambda \underline{a}, \underline{y}, \underline{u}, \underline{x}, \underline{v}.\, x \\ t_V & :\equiv & \Pi = \lambda \underline{a}, \underline{y}, \underline{u}, \underline{x}, \underline{v}.\, v \\ t_Y & :\equiv & \Pi = \lambda \underline{a}, \underline{y}, \underline{u}.\, y \\ t_U & :\equiv & \Pi = \lambda \underline{a}, \underline{y}, \underline{u}.\, u \end{array} \right.$

Recent BRICS Report Series Publications

RS-03-12 **Mircea-Dan Hernest and Ulrich Kohlenbach.** ***A Complexity Analysis of Functional Interpretations*****. February 2003. 70 pp.**

RS-03-11 **Mads Sig Ager, Olivier Danvy, and Henning Korsholm Rohde.** ***Fast Partial Evaluation of Pattern Matching in Strings*****. February 2003. 14 pp.**

RS-03-10 **Federico Crazzolara and Giuseppe Milicia.** ***Wireless Authentication in χ-Spaces*****. February 2003. 20 pp.**

RS-03-9 **Ivan B. Damgård and Gudmund Skovbjerg Frandsen.** ***An Extended Quadratic Frobenius Primality Test with Average and Worst Case Error Estimates*****. February 2003.**

RS-03-8 **Ivan B. Damgård and Gudmund Skovbjerg Frandsen.** ***Efficient Algorithms for gcd and Cubic Residuosity in the Ring of Eisenstein Integers*****. February 2003.**

RS-03-7 **Claus Brabrand, Michael I. Schwartzbach, and Mads Vanggaard.** ***The METAFRONT System: Extensible Parsing and Transformation*****. February 2003. 24 pp.**

RS-03-6 **Giuseppe Milicia and Vladimiro Sassone.** ***Jeeg: Temporal Constraints for the Synchronization of Concurrent Objects*****. February 2003. 41 pp. Short version appears in Fox and Getov, editors,** ***Joint ACM-ISCOPE Conference on Java Grande*****, JGI '02 Proceedings, 2002, pages 212–221.**

RS-03-5 **Aske Simon Christensen, Anders Møller, and Michael I. Schwartzbach.** ***Precise Analysis of String Expressions*****. February 2003. 15 pp.**

RS-03-4 **Marco Carbone and Mogens Nielsen.** ***Towards a Formal Model for Trust*****. January 2003.**

RS-03-3 **Claude Crépeau, Paul Dumais, Dominic Mayers, and Louis Salvail.** ***On the Computational Collapse of Quantum Information*****. January 2003. 31 pp.**

RS-03-2 **Olivier Danvy and Pablo E. Martínez López.** ***Tagging, Encoding, and Jones Optimality*****. January 2003. To appear in Degano, editor,** ***Programming Languages and Systems: Twelfth European Symposium on Programming*****, ESOP '03 Proceedings, LNCS, 2003.**