The IO and OI hierarchies revisited

Gregory Kobele, Sylvain Salvati

The IO and OI hierarchies revisited

Gregory M. Kobele[1⋆] and Sylvain Salvati[2⋆⋆]

[1] University of Chicago kobele@uchicago.edu
[2] INRIA, LaBRI, Université de Bordeaux sylvain.salvati@labri.fr

Abstract. We study languages of λ-terms generated by IO and OI unsafe grammars. These languages can be used to model meaning representations in the formal semantics of natural languages following the tradition of Montague [19]. Using techniques pertaining to the denotational semantics of the simply typed λ-calculus, we show that the emptiness and membership problems for both types of grammars are decidable. In the course of the proof of the decidability results for OI, we identify a decidable variant of the λ-definability problem, and prove a stronger form of Statman's finite completeness Theorem [28].

1 Introduction

In the end of the sixties, similar but independent lines of research were pursued in formal language theory and in the formal semantics of natural language. Formal language theory was refining the Chomsky hierarchy so as to find an adequate syntactic model of programming languages lying in between the context-free and context-sensitive languages. Among others, this period resulted in the definition of *IO and OI macro languages* by Fischer [12] and the notion of *indexed languages* (which coincide with OI macro languages) by Aho [2]. At the same time, Richard Montague [19] was proposing a systematic way of mapping natural language sentences to logical formulae representing their meanings, providing thereby a solid foundation for the field of formal semantics. The main idea behind these two lines of research can be summed up in the phrase '*going higher-order.*' For macro and indexed grammars, this consisted in parameterizing non-terminals with strings and indices (stacks) respectively, and in Montague's work it consisted in using the simply typed λ-calculus to map syntactic structures to their meanings. Montague was ahead of the formal language theory community which took another decade to go higher-order with the work of Damm [7]. However, the way Damm defined higher-order grammars used (implicitly) a restricted version of the λ-calculus that is now known as the *safe λ-calculus*. This restriction was made explicit by Knapik *et al.* [16] and further studied by Blum and Ong [4]. For formal grammars this restriction has first been lifted by de Groote [8] and Muskens [21] in the context of computational linguistics and as a way of applying Montague's techniques to syntactic modeling.

⋆ The first author was funded by LaBRI while working on this research.
⋆⋆ This work has been supported by ANR-12-CORD-0004 POLYMNIE

In the context of higher-order recursive schemes, Ong showed that safety was not a necessary condition for the decidability of the MSO model checking problem. Moreover, the safety restriction has been shown to be a real restriction by Parys [23]. Nevertheless, concerning the IO and OI hierarchies, the question as to whether safety is a genuine restriction in terms of the definable languages is still an open problem. Aehlig *et al.* [1] showed that for second order OI grammars safety was in fact *not* a restriction. It is nevertheless generally conjectured that for higher-order grammars safety is in fact a restriction.

As we wish to extend Montague's technique with the OI hierarchy so as to enrich it with fixed-point computation as proposed by Moschovakis [20], or as in proposals to handle presuppositions in natural languages by Lebedeva and de Groote [10,9,17], we work with languages of λ-terms rather than with just languages of strings or trees. In the context of languages of λ-terms, safety clearly appears to be a restriction since, as shown by Blum and Ong [4], not every λ-term is safe. Moreover the terms generated by Montague's technique appear to be unsafe in general.

This paper is thus studying the formal properties of the unsafe IO and OI languages of λ-terms. A first property that the use of unsafe grammars brings into the picture is that the unsafe IO hierarchy is strictly included within the unsafe OI hierarchy. The inclusion can be easily shown using a standard CPS transform on the grammars and its strictness is implied by decidability results. Nevertheless, it is worth noting that such a transform cannot be performed on safe grammars, so that it is unclear whether safe IO languages are safe OI languages. This paper focuses primarily on the emptiness and the membership problems for unsafe IO and OI languages, by using simple techniques related to the denotational semantics of the λ-calculus. For the IO case, we are going to recast some known results from Salvati [25,24], so as to emphasize that they derive from the fact that for an IO language and a finite model of the λ-calculus, one can effectively compute the elements of the model which are the interpretations of terms in the language. This allows us to show that the emptiness problem is decidable, and also, using Statman's finite completeness theorem [28], to show that the membership problem is decidable. In contrast to the case for IO languages, we show that this property does not hold for OI languages. Indeed, we prove that the set of closed terms of a given type is an OI language, and thus, since λ-definability is undecidable [18], the set of elements in a finite model that are the interpretation of terms in an OI language cannot be effectively computed. To show the decidability of emptiness and of the membership problems for OI, we prove a theorem that we call the *Observability Theorem*; it characterizes some semantic properties of the elements of an OI language in monotonic models, and leads directly to the decidability of the emptiness problem. For the membership problem we prove a generalization of Statman's finite completeness theorem which, in combination with the Observability Theorem, entails the decidability of the membership problem of OI languages.

This work is closely related to the research that is being carried out on higher-order recursive schemes. It differs from it in one important respect: the

main objects of study in the research on higher-order recursive schemes are the infinite trees generated by schemes, while our work is related to the study of the Böhm trees of λY-terms which may contain λ-binders. Such Böhm trees are closer to the configuration graphs of Higher-order Collapsible Pushdown Automata whose first-order theory has been shown undecidable [6]. If we were only interested in grammars generating trees or strings, the decidability of MSO for higher-order recursion schemes [22] would yield the decidability of both the emptiness and the membership problems of unsafe OI grammars, but this is no longer the case when we turn to languages of λ-terms.

Organization of the paper We start by giving the definitions related to the λ-calculus, its finitary semantics, and how to define higher-order grammars in section 2. We then present the decidability results concerning higher-order IO languages and explain why the techniques used there cannot be extended to OI languages in section 3. Section 4 contains the main contributions of the paper: the notion of hereditary prime elements of monotone models together with the Observability Theorem, and a strong form of Statman's finite completeness Theorem. Finally we present conclusions and a broader perspective on our results in section 5. Most of the proofs of our results are in the appendix at the end of the paper.

2 Preliminaries

In this section, we introduce the various calculi we are going to use in the course of the article. Then we show how those calculi may be used to define IO and OI grammars. We give two presentations of those grammars, one using traditional rewriting systems incorporating non-terminals, and the other as terms in one of the calculi; these two perspectives are equivalent. In the remainder of the paper we will switch between these two formats as is most convenient. Finally we introduce the usual notions of full and monotone models for the calculi we work with.

2.1 λ-calculi

We introduce here various extensions of the simply typed λ-calculus. Given an atomic type 0 (our results extend with no difficulty to arbitrarily many atomic types), the set *type* of types is built inductively using the binary right-associative infix operator $\to$. We write $\alpha_1 \to \cdots \to \alpha_n \to \alpha_0$ for $(\alpha_1 \to (\cdots(\alpha_n \to \alpha_0)))$. As in [14], the order of a type is: $\mathrm{order}(0) = 1$, $\mathrm{order}(\alpha \to \beta) = \max(\mathrm{order}(\alpha) + 1, \mathrm{order}(\beta))$. Constants are declared in higher-order signatures Σ which are finite sets of typed constants $\{A_1^{\alpha_1}, \ldots, A_n^{\alpha_n}\}$. We use constants to represent non-terminal symbols.

We assume that we are given a countably infinite set of typed λ-variables (x^α, y^β,...). The families of typed $\lambda Y{+}\Omega$-terms $(\Lambda^\alpha)_{\alpha \in type}$ built on a signature Σ are inductively constructed according to the following rules: x^α, c^α and Ω^α

are in Λ^α; Y^α is in $\Lambda^{(\alpha\to\alpha)\to\alpha}$; if M is in $\Lambda^{\alpha\to\beta}$ and N is in Λ^α, then (MN) is in Λ^α; if M is in Λ^β then $(\lambda x^\alpha.M)$ is in $\Lambda^{\alpha\to\beta}$; if M and N are in Λ^0 then $M+N$ is in Λ^0. When M is in Λ^α we say that it has type α, we write M^α to indicate that M has type α; the order of a term M, is the order of its type. As it is customary, we omit type annotations when they can be easily inferred or when they are irrelevant. We adopt the usual conventions about dropping parentheses in the syntax of terms. We write $M^{\alpha\to\beta}+N^{\alpha\to\beta}$ as an abbreviation for $\lambda x^\alpha.Mx+Nx$. The set of free variables of the term M is written $FV(M)$; a term M is closed when $FV(M)=\emptyset$. Finally we write $M[x_1\leftarrow N_1,\ldots,x_n\leftarrow N_n]$ for the simultaneous capture-avoiding substitutions of the terms N_1, ..., N_n for the free occurrences of the variables x_1, ..., x_n in M.

The set of λ-terms is the set of terms that do not contain occurrences of Y, $+$ or Ω, and for any $S\subseteq\{Y,+,\Omega\}$, the λS-terms are the λ-terms that may contain only constants that are in S. For example, $\lambda{+}\Omega$-terms are the terms that do not contain occurrences of Y.

We assume the reader is familiar with the notions of β-contractions and η-contraction and η-long forms (see [14]). The constant Ω^α stands for the undefined term of type α, Y^α is the fixpoint combinator of type $(\alpha\to\alpha)\to\alpha$, and $+$ is the non-deterministic choice operator. The families of terms that may contain occurrences of Ω are naturally ordered with the least compatible relation $\sqsubseteq$ such that $\Omega^\alpha\sqsubseteq M$ for every term M of type α; δ-contraction provides the operational semantics of the fixpoint combinator: $YM\to_\delta M(YM)$, and $+$-contraction gives the operational semantics of the non-deterministic choice operator: $M+N\to_+ M$ and $M+N\to_+ N$. Given a set $\mathcal{R}$ of symbols denoting compatible relations, for $S\subseteq\mathcal{R}$, S-contraction is the union of the contraction relations denoted by the symbols in S; it will generally be written as $\to_S$. For example, $\to_{\beta\eta+}$ denotes the $\beta\eta+$-contraction. S-reduction, written $\overset{*}{\to}_S$, is the reflexive transitive closure of S-contraction and S-conversion, $=_S$, is the smallest equivalence relation containing $\to_S$. The notion of S-normal form is defined as usual, we simply say *normal form* when S is obvious from the context. We recall (see [14]) that when $M\overset{*}{\to}_{\beta\eta\delta+}N$ and M' and N' are respectively the η-long forms of M and N, then $M'\overset{*}{\to}_{\beta\delta+}N'$. In the remainder of the paper we assume that we are working with terms in η-long form and forget about η-reduction.

2.2 IO/OI grammars and $\lambda Y{+}$-calculus

We define a higher-order macro grammar $\mathcal{G}$ as a triple (Σ,R,S) where Σ is a higher-order signature of non-terminals, R is a finite set of rules $A\to M$ where A is a non-terminal of Σ and M is a λ-term built on Σ that has the same type as A, and where S is a distinguished non-terminal of Σ, the *start symbol*. We do not require M to be a closed term, the free variables in the right hand side of grammatical rules play the role of terminal symbols. We also do not require S to be of a particular type, this permits (higher-order) macro grammars to define languages of λ-terms of arbitrary types. As noted in [8], languages of strings and trees are particular cases of the languages we study. A grammar

has order n when the highest order of its non-terminals is n. Our higher-order macro grammars generalize those of Damm [7] in two ways: first, they do not necessarily verify the safety condition that Damm's grammar implicitly verify; second, instead of only defining languages of strings or trees, they can define languages of λ-terms, following Montague's tradition in the formal semantics of natural languages. According to [4], a term M is safe when no subterm N of M contains free variables (excluding the free variables of M which play the role of constants) of order lower than that of N, unless N occurs as part of a subterm NP or $\lambda x.N$. A grammar is safe when the right hand sides of its rules are all safe terms. Safe terms can be *safely* reduced using substitution in place of capture-avoiding substitutions.

The rules of a grammar $\mathcal{G} = (\Sigma, R, S)$ define a natural relation $\rightarrow_{\mathcal{G}}$ on terms built on Σ. We write $M \rightarrow_{\mathcal{G}} N$ when N is obtained from M by replacing (without capturing free variables) an occurrence of a non-terminal A in M by a term P, such that $A \rightarrow P$ is a rule of $\mathcal{G}$. The grammar $\mathcal{G}$ defines two languages: $\mathcal{L}_{OI}(\mathcal{G}) = \{M \text{ in normal form} \mid S \stackrel{*}{\rightarrow}_{\beta\mathcal{G}} M\}$ and $\mathcal{L}_{IO}(\mathcal{G}) = \{M \text{ in normal form} \mid \exists P. S \stackrel{*}{\rightarrow}_{\mathcal{G}} P \wedge P \stackrel{*}{\rightarrow}_{\beta} M\}$. These two languages can be defined in a different manner, in particular M is in $\mathcal{L}_{OI}(\mathcal{G})$ iff S can be reduced to M with the head reduction strategy that consists in always contracting top-most redices of the relation $\rightarrow_{\beta\mathcal{G}}$. For a given grammar $\mathcal{G}$, we always have that $\mathcal{L}_{IO}(\mathcal{G}) \subseteq \mathcal{L}_{OI}(\mathcal{G})$, but, in general, $\mathcal{L}_{IO}(\mathcal{G}) \neq \mathcal{L}_{OI}(\mathcal{G})$. Here follows an example of a second order macro grammar $\mathcal{G}$ whose free variables (or terminals) are $\textsc{ex}^{(0\rightarrow 0)\rightarrow 0}$, $\textsc{and}^{0\rightarrow 0\rightarrow 0}$, $\textsc{not}^{0\rightarrow 0}$ and $\mathrm{P}^{0\rightarrow 0}$ and whose non-terminals are S^0 (the start symbol), $S'^{0\rightarrow 0}$ and $\mathbf{cons}^{0\rightarrow 0\rightarrow 0}$ (extending BNF notation to macro grammars).

$$
\begin{aligned}
S &\rightarrow \textsc{ex}(\lambda x.(S' x)) \mid \textsc{and}\, S\, S \mid \textsc{not} S \\
S' &\rightarrow \lambda y.\textsc{ex}(\lambda x.S'(\mathbf{cons}\, y\, x)) \mid \lambda y.\textsc{and}(S' y)(S' y) \mid \lambda y.\textsc{not}(S' y) \mid \lambda y.\mathrm{P} y \\
\mathbf{cons} &\rightarrow \lambda xy.x \mid \lambda xy.y
\end{aligned}
$$

The language $\mathcal{L}_{OI}(\mathcal{G})$ represents the set of formulae of first-order logic built with one predicate P (we use a similar construction later on to prove Theorem 8). The language $\mathcal{L}_{IO}(\mathcal{G})$ represents the formulae of first-order logic that can be built with only one variable (that is each subformula of a formula represented in $\mathcal{L}_{IO}(\mathcal{G})$ contains at most one free variable). Given the definition of safety given in [4], it is easily verified that the terms of these languages are not safe; this illustrates that unsafe IO and OI languages of λ-terms are more general than their safe counterparts. Moreover, when seen as graphs, the terms of $\mathcal{L}_{OI}(\mathcal{G})$ form a class of graphs which has an unbounded treewidth; the MSO theory of these terms is undecidable. This explains why the decidability results we obtain later on cannot be seen as corollaries of Ong's Theorem [22].

We extend the notions of IO and OI languages to $\lambda Y{+}\Omega$-terms. Given a $\lambda Y{+}\Omega$-term M, its IO language $\mathcal{L}_{IO}(M)$ is the set of λ-terms N in normal form such that there is P such that $M \stackrel{*}{\rightarrow}_{\delta+} P \stackrel{*}{\rightarrow}_{\beta} N$. Its OI language $\mathcal{L}_{OI}(M)$ is the set of λ-terms N in normal form such that $M \stackrel{*}{\rightarrow}_{\beta\delta+} N$ (we can also restrict our attention to head-reduction). An alternative characterization of $\mathcal{L}_{OI}(M)$ is the

following. Given a term M we write $\omega(M)$ for the *immediate approximation of* M, that is the term obtained from M as follows: $\omega(\lambda x^\alpha.M) = \Omega^{\alpha\to\beta}$ if $\omega(M) = \Omega^\beta$, and $\lambda x^\alpha.\omega(M)$ otherwise; $\omega(MN) = \Omega^\beta$ if $\omega(M) = \Omega^{\alpha\to\beta}$ or $M = \lambda x.P$, and $\omega(MN) = \omega(M)\omega(N)$ otherwise; $\omega(Y^\alpha) = \Omega^{(\alpha\to\alpha)\to\alpha}$, $\omega(x^\alpha) = x^\alpha$, $\omega(\Omega^\alpha) = \Omega^\alpha$, and $\omega(N_1 + N_2) = \omega(N_1) + \omega(N_2)$. Note that $\omega(M)$ is a $\lambda{+}\Omega$-term that contains no β-redices. A $\lambda{+}\Omega$-term Q is a *finite approximation of* M if there is a P such that $M \stackrel{*}{\to}_{\beta\delta} P$ and $Q = \omega(P)$. The language $\mathcal{L}_{OI}(M)$ is the union of the languages $\mathcal{L}_{OI}(Q)$ so that Q is a finite approximation of M.

In both the IO and OI mode of evaluation, $\lambda{+}\Omega$-terms define finite languages, and $\lambda Y{+}\Omega$-calculus defines exactly the same classes of languages as higher-order macro grammars.

Theorem 1. *Given a higher-order macro grammar $\mathcal{G}$, there is a $\lambda Y{+}\Omega$-term M so that $\mathcal{L}_{OI}(\mathcal{G}) = \mathcal{L}_{OI}(M)$ and $\mathcal{L}_{IO}(\mathcal{G}) = \mathcal{L}_{IO}(M)$.*

Given a $\lambda Y{+}\Omega$-term M there is a higher-order macro grammar $\mathcal{G}$ so that $\mathcal{L}_{OI}(\mathcal{G}) = \mathcal{L}_{OI}(M)$ and $\mathcal{L}_{IO}(\mathcal{G}) = \mathcal{L}_{IO}(M)$.

The proof of this theorem is based on the correspondence between higher-order schemes and λY-calculus that is given in [27]. Going from a $\lambda Y{+}\Omega$-term to a grammar is simply a direct transposition of the procedure described in [27] with the obvious treatment for $+$. For the other direction, it suffices to see the grammar as a non-deterministic scheme, which is done by viewing all the rules $A \to M_1, \ldots, A \to M_n$, of a non-terminal A as a unique rule of a scheme $A \to M_1 + \cdots + M_n$; and then to transform the scheme into a $\lambda Y{+}$-term using the transformation given in [27]. There is a minor technicality concerning the IO languages; one needs to start with a grammar where every non-terminal can be rewritten into a $\mathcal{G}$-normal form using $\stackrel{*}{\to}_{\mathcal{G}}$ only.

2.3 Models of the λ-calculi

Full models of the λ-calculus We start by giving the simplest notion of models of λ-calculus, that of full models. A full model $\mathcal{F}$ is a collection of sets indexed by types $(\mathcal{F}_\alpha)_{\alpha\in type}$ so that $\mathcal{F}_{\alpha\to\beta}$ is $\mathcal{F}_\beta^{\mathcal{F}_\alpha}$, the set of functions from $\mathcal{F}_\alpha$ to $\mathcal{F}_\beta$. Note that $\mathcal{F}$ is completely determined by $\mathcal{F}_0$. A full model is said to be finite when $\mathcal{F}_0$ is a finite set; in that case $\mathcal{F}_\alpha$ is finite for every $\alpha \in type$. A valuation ν is a function that maps variables to elements of $\mathcal{F}$ respecting typing, meaning that, for every x^α, $\nu(x^\alpha)$ is in $\mathcal{F}_\alpha$. Given a valuation ν and a in $\mathcal{F}_\alpha$, we write $\nu[x^\alpha \leftarrow a]$ for the valuation which maps the variable x^α to a but is otherwise equal to ν. We can now interpret λ-terms in $\mathcal{F}$, using the following interpretation scheme: $[\![x^\alpha]\!]^\nu_{\mathcal{F}} = \nu(x^\alpha)$, $[\![MN]\!]^\nu_{\mathcal{F}} = [\![M]\!]^\nu_{\mathcal{F}}([\![N]\!]^\nu_{\mathcal{F}})$ and for a in $\mathcal{F}_\alpha$, $[\![\lambda x^\alpha.M]\!]^\nu_{\mathcal{F}}(a) = [\![M]\!]^{\nu[x^\alpha\leftarrow a]}_{\mathcal{F}}$. For a closed term M, $[\![M]\!]^\nu_{\mathcal{F}}$ does not depend on ν, and thus we simply write $[\![M]\!]_{\mathcal{F}}$. The following facts are known about full models:

Theorem 2. *If $M =_{\beta\eta} N$ then for every full model $\mathcal{F}$, $[\![M]\!]^\nu_{\mathcal{F}} = [\![N]\!]^\nu_{\mathcal{F}}$.*

Theorem 3 (Finite Completeness [28]). *Given a λ-term M, there is a finite full model $\mathcal{F}_M$ and a valuation ν so that, for every N, $[\![N]\!]^\nu_{\mathcal{F}_M} = [\![M]\!]^\nu_{\mathcal{F}_M}$ iff $N =_{\beta\eta} M$.*

In this theorem, the construction of $\mathcal{F}_M$ and ν is effective.

For a full model $\mathcal{F}$, an element f of $\mathcal{F}_\alpha$ is said to be λ-definable when there is a closed M such that $[\![M]\!]_{\mathcal{F}} = f$. The problem of λ-definability is the problem whose input is a finite full model $\mathcal{F}$ and an element f of $\mathcal{F}_\alpha$, and whose answer is whether f is λ-definable.

Theorem 4 (Loader [18]). *The λ-definability problem is undecidable.*

Given a language of λ-terms of type α, L, and a full model $\mathcal{F}$, we write $[\![L]\!]^\nu_{\mathcal{F}}$ for the set $\{[\![M]\!]^\nu_{\mathcal{F}} \mid M \in L\}$. So in particular, for a $\lambda Y{+}\Omega$-term M, we may write $[\![\mathcal{L}_{IO}(M)]\!]^\nu_{\mathcal{F}}$ or $[\![\mathcal{L}_{OI}(M)]\!]^\nu_{\mathcal{F}}$.

Monotone models of $\lambda Y{+}\Omega$-calculus Given two complete lattices L_1 and L_2, we write $L_3 = \mathbf{Mon}[L_1 \to L_2]$ for the lattice of monotonous functions from L_1 to L_2 ordered pointwise; f is monotonous if $a \leq_1 b$ implies $f(a) \leq_2 f(b)$, and given f and g in L_3, $f \leq_3 g$ whenever for every a in L_1, $f(a) \leq_2 g(a)$. Among the functions in $\mathbf{Mon}[L_1 \to L_2]$, of special interest are the step functions which are functions $a \mapsto f$ determined from elements a in L_1 and f in L_2, and are defined such that $(a \mapsto f)(b)$ is equal to f when $a \leq_1 b$ and to $\bot_2$ otherwise. A monotone model $\mathcal{M}$ is a collection of finite lattices indexed by types, $(\mathcal{M}_\alpha)_{\alpha \in type}$ where $\mathcal{M}_{\alpha\to\beta} = \mathbf{Mon}[\mathcal{M}_\alpha \to \mathcal{M}_\beta]$ (we write $\bot_\alpha$ and $\top_\alpha$ respectively for the least and greatest elements of $\mathcal{M}_\alpha$). The notion of valuation on monotone models is similar to the one on full models and we use the same notation. Terms are interpreted in monotone models according to the following scheme: $[\![x^\alpha]\!]^\nu_{\mathcal{M}} = \nu(x^\alpha)$, $[\![MN]\!]^\nu_{\mathcal{M}} = [\![M]\!]^\nu_{\mathcal{M}}([\![N]\!]^\nu_{\mathcal{M}})$ and for a in $\mathcal{M}_\alpha$, $[\![\lambda x^\alpha.M]\!]^\nu_{\mathcal{M}}(a) = [\![M]\!]^{\nu[x^\alpha \leftarrow a]}_{\mathcal{M}}$, $[\![\Omega^\alpha]\!]^\nu_{\mathcal{M}} = \bot_\alpha$, $[\![M+N]\!]^\nu_{\mathcal{M}} = [\![M]\!]^\nu_{\mathcal{M}} \vee [\![N]\!]^\nu_{\mathcal{M}}$ and for every a in $\mathcal{M}_{\alpha\to\alpha}$, $[\![Y]\!]^\nu_{\mathcal{M}}(a) = \bigvee\{a^n(\bot_\alpha) \mid n \in \mathbb{N}\}$.

The following Theorem gives well known results on monotone models (see [3]):

Theorem 5. *Given two $\lambda Y{+}\Omega$terms of type α, M and N:*

1. *if $M =_{\beta\delta} N$ then for every monotone model, $\mathcal{M}$, $[\![M]\!]^\nu_{\mathcal{M}} = [\![N]\!]^\nu_{\mathcal{M}}$,*
2. *$[\![M]\!]^\nu_{\mathcal{M}} = \bigvee\{[\![Q]\!]^\nu_{\mathcal{M}} \mid Q$ is a finite approximation of $M\}$,*
3. *if $M \xrightarrow{*}_{\beta\delta+} N$ then $[\![N]\!]^\nu_{\mathcal{M}} \leq [\![M]\!]^\nu_{\mathcal{M}}$,*
4. *if $N \sqsubseteq M$ then $[\![N]\!]^\nu_{\mathcal{M}} \leq [\![M]\!]^\nu_{\mathcal{M}}$.*

3 Relations between IO, OI and full models

In this section, we investigate some basic properties of IO and OI languages. We will see that the class of higher-order OI languages strictly subsumes that of higher-order IO languages. We will then see that the emptiness and membership problems for higher-order IO languages are decidable, by showing that

for a higher-order grammar $\mathcal{G}$, a finite full model $\mathcal{F}$, and a valuation ν, the set $[\![\mathcal{L}_{IO}(\mathcal{G})]\!]^{\nu}_{\mathcal{F}}$ is effectively computable. On the other hand, we show that $[\![L]\!]^{\nu}_{\mathcal{F}}$ is *not* in general effectively computable when L is an OI language.

A simple continuation passing style (CPS) transform witnesses that:

Theorem 6 (OI subsumes IO). *Given a higher-order grammar $\mathcal{G}$ there is a higher-order grammar $\mathcal{G}'$ so that $\mathcal{L}_{IO}(\mathcal{G}) = \mathcal{L}_{OI}(\mathcal{G}')$.*

The CPS transform naturally makes the order of $\mathcal{G}'$ be the order of $\mathcal{G}$ plus 2.

We now show that for a full model $\mathcal{F}$, a valuation ν and a given grammar $\mathcal{G}$, the set $[\![\mathcal{L}_{IO}(\mathcal{G})]\!]^{\nu}_{\mathcal{F}}$ can be effectively computed. A natural consequence of this is that the emptiness and the membership problems for higher-order IO languages are decidable. These results are known in the literature [24,25,26], nevertheless, we include them here so as to emphasize that they are related to the effectivity of the set $[\![\mathcal{L}_{IO}(\mathcal{G})]\!]^{\nu}_{\mathcal{F}}$, a property that, as we will see later, does not hold in the case of OI languages.

Theorem 7 (Effective finite interpretation of IO). *Given a higher-order macro grammar $\mathcal{G}$, a full model $\mathcal{F}$ and a valuation ν, one can effectively construct the set $[\![\mathcal{L}_{IO}(\mathcal{G})]\!]^{\nu}_{\mathcal{F}}$.*

Corollary 1. *Given a higher-order macro grammar $\mathcal{G}$, the problem of deciding whether $\mathcal{L}_{IO}(\mathcal{G}) = \emptyset$ is P-complete.*

Corollary 2. *Given a higher-order macro grammar $\mathcal{G}$ and a term M, it is decidable whether $M \in \mathcal{L}_{IO}(\mathcal{G})$.*

We are now going to see that the set of closed λ-terms of a given type α is an OI language. Combined with Theorem 4, we obtain that the set $[\![\mathcal{L}_{OI}(\mathcal{G})]\!]_{\mathcal{F}}$ cannot be effectively computed. Moreover, Theorems 6 and 7 imply that the IO hierarchy is strictly included in the OI hierarchy.

Theorem 8. *For every type α, there is a closed $\lambda Y+$-term M of type α such that $\mathcal{L}_{OI}(M)$ is the set of all closed normal λ-terms of type α.*

Theorem 9 (Undecidable finite interpretation of OI). *Given a higher-order macro grammar $\mathcal{G}$, a finite full model $\mathcal{F}$, and f an element of $\mathcal{F}$, it is undecidable whether $f \in [\![\mathcal{L}_{OI}(\mathcal{G})]\!]$.*

Proof. Direct consequence of Theorems 8 and 4. □

Theorem 10. *The class of higher-order IO language is strictly included in the class of higher-order OI languages.*

Proof. If there were an IO grammar that could define the set of closed terms of type α, Theorem 7 would contradict Theorem 4. □

This last theorem should be contrasted with the result of Haddad [13] which shows that OI and IO coincide for schemes. The two results do not contradict each other as IO is not defined in the same way on schemes and on grammars.

4 Emptiness and membership for the OI hierarchy

In this section we prove the decidability of the emptiness and membership problems for higher-order OI languages. For this we use monotone models as approximations of sets of elements of full models.

4.1 Hereditary primality and the Observability Theorem

Theorem 9 implies that the decision techniques we used for the emptiness and the membership problems for IO do not extend to OI. So as to show that those problems are nevertheless decidable, we are going to prove a theorem that we call the *Observability Theorem*, which allows us to *observe* certain semantic properties of λ-terms in the OI language of a $\lambda Y{+}\Omega$-term M by means of the semantic values of M in monotone models. For this we introduce the notion of *hereditary prime elements* of a monotone model.

Definition 1. *In a lattice $\mathcal{L}$, an element f is* prime *(or* $\vee$-prime*) when for every g_1 and g_2 in $\mathcal{L}$, $f \leq g_1 \vee g_2$ implies that $f \leq g_1$ or $f \leq g_2$.*

Given a monotone model $\mathcal{M} = (\mathcal{M}_\alpha)_{\alpha \in type}$, for every type α we define the sets $\mathcal{M}^+_\alpha$ and $\mathcal{M}^-_\alpha$ by:

1. *$\mathcal{M}^+_0$ and $\mathcal{M}^-_0$ contain the prime elements of $\mathcal{M}_0$ that are different from $\bot_0$,*
2. *$\mathcal{M}^+_{\alpha\to\beta} = \{(\bigvee F) \mapsto g \mid F \subseteq \mathcal{M}^-_\alpha \wedge g \in \mathcal{M}^+_\beta\}$,*
3. *$\mathcal{M}^-_{\alpha\to\beta} = \{f \mapsto g \mid f \in \mathcal{M}^+_\alpha \wedge g \in \mathcal{M}^-_\beta\}$.*

A valuation ν on $\mathcal{M}$ is said hereditary prime *when, for every variable x^α, $\nu(x^\alpha) = \bigvee F$ for some $F \subseteq \mathcal{M}^-_\alpha$. The elements of $\mathcal{M}^+_\alpha$ are called the* hereditary prime elements *of $\mathcal{M}_\alpha$.*

The main interest of primality lies in that, if f is prime and $f \leq [\![M+N]\!]^\nu_{\mathcal{M}}$, then either $f \leq [\![M]\!]^\nu_{\mathcal{M}}$ or $f \leq [\![N]\!]^\nu_{\mathcal{M}}$. The notion of hereditary primality is simply a way of making primality compatible with all the constructs of $\lambda Y{+}\Omega$-terms. The proof of the following technical Lemma from which we derive the Observability Theorem, is mainly based on this idea.

Lemma 1. *Given a $\lambda{+}\Omega$-term M^α, a monotone model $\mathcal{M} = (\mathcal{M}_\alpha)_{\alpha\in type}$, a hereditary prime valuation ν and a hereditary prime element f of $\mathcal{M}_\alpha$, we have the equivalence:*

$$f \leq [\![M^\alpha]\!]^\nu_{\mathcal{M}} \Leftrightarrow \exists N \in \mathcal{L}_{OI}(M).f \leq [\![N]\!]^\nu_{\mathcal{M}}$$

Theorem 5 allows to extend Lemma 1 to $\lambda Y{+}\Omega$-terms.

Theorem 11 (Observability). *Given a $\lambda Y{+}\Omega$-term M, a monotone model $\mathcal{M} = (\mathcal{M}_\alpha)_{\alpha\in type}$, a hereditary prime valuation ν and a hereditary prime element f of $\mathcal{M}_\alpha$, we have the equivalence:*

$$f \leq [\![M^\alpha]\!]^\nu_{\mathcal{M}} \Leftrightarrow \exists N \in \mathcal{L}_{OI}(M).f \leq [\![N]\!]^\nu_{\mathcal{M}}$$

Proof. Since for every α, $\mathcal{M}_\alpha$ is finite, according to Theorem 5.2 (and the fact that the set of finite approximations of M is directed for the partial order $\sqsubseteq$), there is a finite approximation Q of M such that $[\![Q]\!]^\nu_{\mathcal{M}} = [\![M]\!]^\nu_{\mathcal{M}}$ and thus $f \leq [\![Q^\alpha]\!]^\nu_{\mathcal{M}}$. But then Q is a λ+Ω-term and by the previous Lemma this is equivalent to there being some N in $\mathcal{L}_{OI}(Q)$ such that $f \leq [\![N]\!]^\nu_{\mathcal{M}}$. The conclusion follows from the fact that obviously $\mathcal{L}_{OI}(Q) \subseteq \mathcal{L}_{OI}(M)$. The other direction follows from Theorem 5.3. □

4.2 Decidability results

We are now going to use the Observability Theorem so as to prove the decidability of both the emptiness and the membership problems for OI languages.

Decidability of emptiness We consider the monotone model $\mathcal{E} = (\mathcal{E}_\alpha)_{\alpha \in type}$ so that $\mathcal{E}_0$ is the lattice with two elements $\{\top, \bot\}$ so that $\bot \leq \top$. We then define for every α, the element $\mathbf{e}_\alpha$ of $\mathcal{E}^+_\alpha \cap \mathcal{E}^-_\alpha$ so that: $\mathbf{e}_0 = \top$, and $\mathbf{e}_{\alpha\to\beta} = \mathbf{e}_\alpha \mapsto \mathbf{e}_\beta$. We let ξ be the valuation so that for each variable x^α, $\xi(x^\alpha) = \mathbf{e}_\alpha$. A simple induction gives the following Lemma which, combined with Theorem 11 gives proposition 1 and finally Theorem 12:

Lemma 2. *For every λ-term M of type γ, $\mathbf{e}_\gamma \leq [\![M]\!]^\xi_{\mathcal{E}}$.*

Proposition 1. *Given a λY+Ω-term M of type α, we have that*

$$\mathcal{L}_{OI}(M) \neq \emptyset \Leftrightarrow \mathbf{e}_\alpha \leq [\![M]\!]^\xi_{\mathcal{E}}$$

Proof. If $\mathcal{L}_{OI}(M) \neq \emptyset$, then there is N in normal form so that $M \xrightarrow{*}_{\beta\delta+} N$. Lemma 2 implies that $\mathbf{e}_\alpha \leq [\![N]\!]^\xi_{\mathcal{E}}$ and thus, using Theorem 5, $\mathbf{e}_\alpha \leq [\![M]\!]^\xi_{\mathcal{E}}$. If $\mathbf{e}_\alpha \leq [\![M]\!]^\xi_{\mathcal{E}}$, since $\mathbf{e}_\alpha$ is in $\mathcal{E}^+_\alpha$, from Theorem 11, there is N in $\mathcal{L}_{OI}(M)$ so that $\mathbf{e}_\alpha \leq [\![N]\!]^\xi_{\mathcal{E}}$; so in particular that $\mathcal{L}_{OI}(M) \neq \emptyset$. □

Theorem 12. *The emptiness problem for OI languages is decidable.*

Decidability of membership For the decidability of the membership problem, we are going to prove a stronger version of Statman's finite completeness Theorem. The proofs based mostly on logical relations (see [3]) can be found in the appendix.

Given a finite set A, we write $\mathcal{M}(A) = (\mathcal{M}_\alpha(A))_{\alpha \in type}$ for the monotone model so that $\mathcal{M}_0(A)$ is the lattice of subsets of A ordered by inclusion. We let $\bot_{A,\alpha}$ be the least element of $\mathcal{M}_\alpha(A)$.

Definition 2. *Given a λ-term M of type α, a triple $T = (A, \nu, f)$, where A is a finite set, ν is a valuation on $\mathcal{M}(A)$ and f is an element of $\mathcal{M}_\alpha(A)$, is* characteristic *of M when:*

1. *for every λ-term N of type α, $M =_\beta N$ iff $f \leq [\![N]\!]^\nu_{\mathcal{M}(A)}$,*

2. *f is a hereditary prime element of $\mathcal{M}_\alpha(A)$ and ν is a hereditary prime valuation.*

The stronger form of Statman finite complete is formulated as:

Theorem 13 (Monotone finite completeness). *For every type α and every pure term M, one can effectively construct a triple T that is characteristic of M.*

Using the Observability Theorem as in the proof of Proposition 1 we obtain:

Theorem 14. *Given a λ-term M in normal form of type α and a $\lambda Y{+}\Omega$-term N of type α, if $T = (A, \nu, f)$ is a characteristic triple of M then $f \leq [\![N]\!]^{\nu}_{\mathcal{M}(A)}$ iff $M \in \mathcal{L}_{OI}(N)$.*

Theorem 15. *Given M a λ-term of type α and N a $\lambda Y{+}\Omega$-term of type α, it is decidable whether $M \in \mathcal{L}_{OI}(N)$.*

5 Conclusion

We have seen how to use models of λ-calculus so as to solve algorithmic questions, namely the emptiness and membership problems, related to the classes of higher-order IO and OI languages of λ-terms. In so doing, we have revisited various questions related to finite models of the λ-calculus. In particular, we have seen that hereditary prime elements, via the Observability Theorem, play a key role in finding effective solutions for higher-order OI languages. In combination with Theorem 8, we obtain that it is decidable whether there is a term M whose interpretation in a monotone model is greater than a given hereditary prime element of that model, which gives a decidability result for a restricted notion of λ-definability. This raises at least two questions: (i) what kind of properties of λ-terms can be captured with hereditary prime elements, (ii) is there a natural extension of this notion that still defines some decidable variant of λ-definability.

On the complexity side, we expect that, using similar techniques as in [29], it might be possible to prove that verifying whether the value of a $\lambda Y{+}\Omega$-term is greater than a hereditary prime element of a monotone model is of the same complexity as the emptiness and membership problems for the safe OI hierarchy which is $(n-2)$-EXPTIME-complete for order n-grammars (see [11], with Huet's convention, the order of grammars is one plus the order of their corresponding higher-order pushdown automaton). Of course, such a high complexity makes the decidability results we obtained of little interest for practical applications in natural language processing. It does however underscore the need to identify linguistically motivated generalizations which point to tractable subclasses of OI grammars [30]. Some restricted classes of IO grammars are known to have low complexity [15,5]. A natural move is to see whether in the OI mode of derivation those grammars still have reasonable complexity for the emptiness and membership problems.

References

1. K. Aehlig, J.G. de Miranda, and C.-H.L. Ong. Safety is not a restriction at level 2 for string languages. In *FOSSACS*, volume 3441 of *LNCS*, pages 490–504, 2005.
2. A. V. Aho. Indexed grammars - an extension of context-free grammars. *J. ACM*, 15(4):647–671, 1968.
3. R. M. Amadio and P-L. Curien. *Domains and Lambda-Calculi*. Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 1998.
4. W. Blum and C.-H.L. Ong. The safe lambda calculus. *Logical Methods in Computer Science*, 5(1:3):1–38, 2009.
5. Pierre Bourreau and Sylvain Salvati. A datalog recognizer for almost affine λ-cfgs. In *MOL 12*, volume 6878 of *LNCS*, pages 21–38. Springer, 2011.
6. Christopher H. Broadbent. The limits of decidability for first order logic on cpda graphs. In *STACS*, pages 589–600, 2012.
7. W. Damm. The IO- and OI-hierarchies. *Theor Comput Sci*, 20:95–207, 1982.
8. P. de Groote. Towards abstract categorial grammars. In ACL, editor, *Proceedings 39th Annual Meeting of ACL*, pages 148–155, 2001.
9. P. de Groote and E. Lebedeva. On the dynamics of proper names. Technical report, INRIA, 2010.
10. P. de Groote and E. Lebedeva. Presupposition accommodation as exception handling. In *SIGDIAL*, pages 71–74. ACL, 2010.
11. Joost Engelfriet. Iterated stack automata and complexity classes. *Inf. Comput.*, 95(1):21–75, 1991.
12. M. J. Fischer. *Grammars with macro-like productions*. PhD thesis, Harvard University, 1968.
13. A. Haddad. IO vs OI in higher-order recursion schemes. In *FICS*, volume 77 of *EPTCS*, pages 23–30, 2012.
14. G. Huet. *Résolution d'équations dans des langages d'ordre 1,2,...,ω*. Thèse de doctorat en sciences mathématiques, Université Paris VII, 1976.
15. M. Kanazawa. Parsing and generation as datalog queries. In *Proceedings of the 45th Annual Meeting of ACL*, pages 176–183. ACL, 2007.
16. T. Knapik, D. Niwinski, and P. Urzyczyn. Higher-order pushdown trees are easy. In *FoSSaCS*, volume 2303 of *LNCS*, pages 205–222, 2002.
17. E. Lebedeva. *Expressing Discourse Dynamics Through Continuations*. PhD thesis, Université de Lorraine, 2012.
18. R. Loader. The undecidability of λ-definability. In *Logic, Meaning and Computation: Essays in memory of Alonzo Church*, pages 331–342. Kluwer, 2001.
19. R. Montague. *Formal Philosophy: Selected Papers of Richard Montague*. Yale University Press, New Haven, CT, 1974.
20. Y. Moschovakis. Sense and denotation as algorithm and value. In *Logic Colloquium'90: ASL Summer Meeting in Helsinki*, volume 2, page 210. Springer, 1993.
21. Reinhard Muskens. Lambda Grammars and the Syntax-Semantics Interface. In *Proceedings of the Thirteenth Amsterdam Colloquium*, pages 150–155, 2001.
22. C.-H.L. Ong. On model-checking trees generated by higher-order recursion schemes. In *LICS*, pages 81–90, 2006.
23. Pawel Parys. On the significance of the collapse operation. In *LICS*, pages 521–530, 2012.
24. S. Salvati. Recognizability in the Simply Typed Lambda-Calculus. In *16th WOLLIC*, volume 5514 of *LNCS*, pages 48–60. Springer, 2009.

25. S. Salvati. On the membership problem for non-linear acgs. *Journal of Logic Language and Information*, 19(2):163–183, 2010.
26. S. Salvati, G. Manzonetto, M. Gehrke, and H. Barendregt. Loader and Urzyczyn are logically related. In *ICALP (2)*, LNCS, pages 364–376, 2012.
27. S. Salvati and I. Walukiewicz. Recursive schemes, Krivine machines, and collapsible pushdown automata. In *RP*, volume 7550 of *LNCS*, pages 6–20, 2012.
28. R. Statman. Completeness, invariance and λ-definability. *Journal of Symbolic Logic*, 47(1):17–26, 1982.
29. K. Terui. Semantic evaluation, intersection types and complexity of simply typed lambda calculus. In *RTA*, pages 323–338, 2012.
30. Iris van Rooij. The tractable cognition thesis. *Cognitive Science*, 32:939–984, 2008.

A Proofs of section 3

Theorem 6. *Given a higher-order grammar $\mathcal{G}$ there is a higher-order grammar $\mathcal{G}'$ so that $\mathcal{L}_{IO}(\mathcal{G}) = \mathcal{L}_{OI}(\mathcal{G}')$.*

Proof. Let us assume that $\mathcal{G} = (\Sigma, R, S)$. Before we start the proof, we adopt the convention that when, for $A_1, \ldots, A_n$ non-terminals, we write $M = P[x_1 \leftarrow A_1, \ldots, x_n \leftarrow A_n]$ for a λ-term M built on Σ, we implicitly assume that P is a λ-term that does not contain any non-terminal and also that the variables x_1, $\ldots$, x_n have exactly one free occurrence in P.

We now define the higher-order macro grammar $\mathcal{G}' = (\Sigma', R', S)$ as follows: for every A^α in Σ, we let $A'^{\alpha'}$ be in Σ' so that $\alpha' = (\alpha \to 0) \to 0$, we also let S be in Σ'. Now for every rule $A \to P[x_1 \leftarrow A_1, \ldots, x_n \leftarrow A_n]$ in R, we let $A' \to \lambda k.A_1(\lambda x_1 \ldots A_n(\lambda x_n.kP)\ldots)$ be in R'; if S has type $\alpha_1 \to \cdots \to \alpha_n \to 0$, we also add to R' the rule $S \to \lambda x_1 \ldots x_n.S'(\lambda P.Px_1 \ldots x_n)$. It now remains to show that $\mathcal{L}_{IO}(\mathcal{G}) = \mathcal{L}_{OI}(\mathcal{G}')$.

We start by showing that $\mathcal{L}_{IO}(\mathcal{G}) \subseteq \mathcal{L}_{OI}(\mathcal{G}')$. For this we show that when, $A \xrightarrow{*}_{\mathcal{G}} M$ with $M = P[x_1 \leftarrow A_1, \ldots, x_n \leftarrow A_n]$, then

$$A' \xrightarrow{*}_{\beta\mathcal{G}'} \lambda k.A'_{\sigma(1)}(\lambda x_{\sigma(1)} \ldots A'_{\sigma(n)}(\lambda x_{\sigma(n)}.kP)\ldots)$$

for some permutation σ of $[1, n]$. This is proved by induction on the reduction $A \xrightarrow{*}_{\mathcal{G}} M$. In case this reduction has length one the conclusion is clear from the definition of $\mathcal{G}'$. Let's assume the reduction has length $m = k+1$ for $k > 0$. This implies that A can be rewritten in k steps into $N = Q[y_1 \leftarrow B_1, \ldots, y_r \leftarrow B_r]$ and that for some i in $[1, r]$, $B_i \to P$ is a rule of $\mathcal{G}$ so that M is obtained from N by rewriting B_i into P. Without loss of generality we assume that $i = 1$ and thus that $M = Q[y_1 \leftarrow P, y_2 \leftarrow B_2 \ldots, y_r \leftarrow B_r]$ and $P = P'[z_1 \leftarrow C_1, \ldots, z_q \leftarrow C_q]$. Therefore we have:

$$M = (Q[y_1 \leftarrow P'])[z_1 \leftarrow C_1, \ldots, z_q \leftarrow C_q, y_2 \leftarrow B_2 \ldots, y_r \leftarrow B_r]$$

By induction hypothesis, A' can be rewritten in the OI mode of evaluation with the grammar $\mathcal{G}'$ to a term $Q' = \lambda k.B'_{\sigma(1)}(\lambda y_{\sigma(1)} \ldots B'_{\sigma(n)}(\lambda y_{\sigma(n)}.kQ))$. Without loss of generality we assume that $\sigma(1) = 1$. By definition of $\mathcal{G}'$, $B'_1 \to \lambda k.C'_1(\lambda z_1 \ldots C'_q(\lambda z_q.kP')\ldots)$. Therefore Q' can be rewritten to

$$\lambda k.(\lambda k.C'_1(\lambda z_1 \ldots C'_q(\lambda z_q.kP')\ldots))(\lambda y_1 \ldots B'_{\sigma(n)}(\lambda y_{\sigma(n)}.kQ))$$

which itself reduces to

$$\lambda k.C'_1(\lambda z_1 \ldots C'_q(\lambda z_q.B'_{\sigma(2)}(\lambda y_2 \ldots B'_{\sigma(n)}(\lambda y_{\sigma(n)}.kQ[y_1 \leftarrow P'])))\ldots))$$

This finally proves the claim.

Now we can use the claim so as to prove the inclusion. If N is in $\mathcal{L}_{IO}(\mathcal{G})$, it means that $S \xrightarrow{*}_{\mathcal{G}} P$ and $P \xrightarrow{*}_{\beta} N$. Thus, using the previous claim we have that $S' \xrightarrow{*}_{\beta\mathcal{G}'} \lambda k.kN$. But then $S \xrightarrow{*}_{\mathcal{G}'} \lambda x_1 \ldots x_n.S'(\lambda Q.Qx_1 \ldots x_n) \xrightarrow{*}_{\beta\mathcal{G}}$, $\lambda x_1 \ldots x_n.(\lambda k.kN)(\lambda Q.Qx_1 \ldots x_n) \xrightarrow{*}_{\beta} N$ so that N is in $\mathcal{L}_{OI}(\mathcal{G}')$.

Let us now prove the converse inclusion: $\mathcal{L}_{OI}(\mathcal{G}') \subseteq \mathcal{L}_{IO}(\mathcal{G})$. Here the trick consists in using head-reduction as the evaluation strategy for OI. We prove that, under this reduction strategy, when a non-terminal A' can be rewritten in $3m+1$ steps into a term M that still contains a non-terminal, then M is of the form $\lambda k.A'_1(\lambda x_1.\dots A'_n(\lambda x_n.kN)\dots)$, and $A \xrightarrow{*}_{\mathcal{G}} N[x_1 \leftarrow A_1, \dots, x_n \leftarrow A_n]$. This claim can be proved by induction on m. The case where $m = 0$ is clear from the definition of $\mathcal{G}'$. If $m = k+1$, then by the induction hypothesis A' can be rewritten in $3k+1$ steps into $\lambda k.A'_1(\lambda x_1.\dots A'_n(\lambda x_n.kN)\dots)$ and $A \xrightarrow{*}_{\mathcal{G}} N[x_1 \leftarrow A_1, \dots, x_n \leftarrow A_n]$. Now there must be a rule $A'_1 \to \lambda k.B'_1(\lambda z_1.\dots B'_p(\lambda z_p.kP))$ in R' so that:

$$\begin{aligned}
&\lambda k.A'_1(\lambda x_1.A'_2(\lambda x_2.\dots A'_n(\lambda x_n.kN)\dots)) \\
&\quad \to_{\mathcal{G}'} \lambda k.(\lambda k.B'_1(\lambda z_1.\dots B'_p(\lambda z_p.kP)))(\lambda x_1.A'_2(\lambda x_2.\dots A'_n(\lambda x_n.kN)\dots)) \\
&\quad \to_{\beta} \lambda k.B'_1(\lambda z_1.\dots B'_p(\lambda z_p.(\lambda x_1.A'_2(\lambda x_2.\dots A'_n(\lambda x_n.kN)\dots)P)\dots)) \\
&\quad \to_{\beta} \lambda k.B'_1(\lambda z_1.\dots B'_p(\lambda z_p.(A'_2\lambda x_2.\dots A'_n(\lambda x_n.kN[x_1 \leftarrow P])\dots))\dots)
\end{aligned}$$

But since $A'_1 \to \lambda k.B'_1(\lambda z_1.\dots B'_p(\lambda z_p.kP))$ is in R' we must have $A_1 \to P[z_1 \leftarrow B_1, \dots, z_p \leftarrow B_p]$ in R and thus $N[x_1 \leftarrow A_1, \dots, x_n \leftarrow A_n]$ can be rewritten into $N[x_1 \leftarrow P[z_1 \leftarrow B_1, \dots, z_p \leftarrow B_p], \dots, x_n \leftarrow A_n]$ that is into $(N[x_1 \leftarrow P])[z_1 \leftarrow B_1, \dots, z_p \leftarrow B_p, x_2 \leftarrow A_2, \dots, x_n \leftarrow A_n]$ and thus the claim is proved by induction.

With the claim we easily obtain that when m is a smallest number so that A' is rewritten into a term M which does not contain any non-terminal in $3m+1$ steps, then $M = \lambda k.kP$ and $A \xrightarrow{*}_{\mathcal{G}} P$. This implies that, if $\mathcal{L}_{IO}(A)$ is the language of terms that A defines in $\mathcal{G}$ and in the IO mode of derivation and $\mathcal{L}_{OI}(A')$ is the language of terms that A' defines in $\mathcal{G}'$ and in the OI mode of derivation, $\lambda k.kN$ is in $\mathcal{L}_{OI}(A')$ implies that N is in $\mathcal{L}_{IO}(A)$. The fact that $\mathcal{L}_{OI}(\mathcal{G}')$ is included into $\mathcal{L}_{IO}(\mathcal{G})$ is an immediate consequence. □

Theorem 7. *Given a higher-order macro grammar $\mathcal{G}$, a full model $\mathcal{F}$ and a valuation ν, one can effectively construct the set $[\![\mathcal{L}_{IO}(\mathcal{G})]\!]^{\nu}_{\mathcal{F}}$.*

Proof. Since this theorem is some reformulation of results that already appeared in the literature, we simply sketch its proof.

The idea behind this theorem is reminiscent of the usual techniques used when facing some problem related to context free grammars. Basically the proof of this Theorem consists in computing a least fixpoint.

For this we assume that $\mathcal{G} = (\Sigma, R, S)$ and that $\mathcal{F} = (\mathcal{F}_\alpha)_{\alpha \in type}$. We define the family $\mathcal{P} = (\mathcal{P}_\alpha)_{\alpha \in type}$ so that $\mathcal{P}_\alpha = 2^{\mathcal{F}_\alpha}$. Given ρ that maps the non-terminals to $\mathcal{F}$ in a type consistent manner (similarly to valuations) we define interpretations, $[\![M]\!]^{\nu,\rho}$ of λ-terms M built on Σ with ρ, and a valuation ν as expected. Now if we let ξ be a type consistent mapping of non-terminals to elements of $\mathcal{P}$, we say that ρ is compatible with ξ relative to the set of non-terminals $\mathcal{N}$, written $\rho \in_{\mathcal{N}} \xi$, when for every A in $\mathcal{N}$, $\rho(A) \in \xi(A)$; for a λ-term M, we write $\rho \in_M \xi$ when ρ is compatible with ξ relative to the non-terminals occurring in M. We then let $[\![M]\!]^{\nu,\xi}_{\mathcal{P}}$ be the set $\{[\![M]\!]^{\nu,\rho}_{\mathcal{F}} \mid \rho \in_M \xi\}$.

We then use this interpretation to compute the least valuation ξ so that for every non-terminal A, if the rules concerning A are $A \to M_1$, …, $A \to$

M_n, then $\xi(A) = \bigcup_{i\in[1,n]} [\![M_i]\!]^{\nu,\xi}_{\mathcal{P}}$. For this, it suffices to let ξ_0 be the valuation of non-terminals so that for every non-terminal A, $\xi_0(A) = \emptyset$, and then let $\xi_{k+1}(A) = \bigcup_{i\in[1,n]} [\![M_i]\!]^{\nu,\xi_k}_{\mathcal{P}}$. It is then easy to see that for every k, and every A, $\xi_k(A) \subseteq \xi_{k+1}(A)$, which guarantees the existence of the least valuation ξ. Then the usual techniques show that an element f of $\mathcal{M}$ is in $\xi(A)$ iff there are P and M so that $A \xrightarrow{*}_{\mathcal{G}} P \xrightarrow{*}_{\beta} M$ and $[\![M]\!]^{\nu}_{\mathcal{F}} = f$. □

Corollary 1. *Given a higher-order macro grammar $\mathcal{G}$, the problem of deciding whether $\mathcal{L}_{IO}(\mathcal{G}) = \emptyset$ is P-complete.*

Proof. The P-hardness of the problem comes from the fact that it subsumes the problem of the emptiness of a context-free language which is P-complete. The fact that it is in P comes from the fact that if we take the full model $\mathcal{S} = (\mathcal{S}_\alpha)_{\alpha\in type}$ so that $\mathcal{S}_0$ is a singleton set, then for every $\alpha \in type$, $\mathcal{S}_\alpha$ is a singleton set. Then $\mathcal{T}_\alpha = 2^{\mathcal{S}_\alpha}$ is a two elements set and computing the interpretation of a term as explained in the proof of Theorem 7 is then linear in the size of the term, so that computing a fixpoint of a grammar in $(\mathcal{T}_\alpha)_{\alpha\in type}$ is linear in the number of non-terminals and in the size of the terms involved in the rules of the grammar. □

Corollary 2. *Given a higher-order macro grammar $\mathcal{G}$, and a term M it is decidable whether $M \in \mathcal{L}_{IO}(\mathcal{G})$.*

Proof. From the finite completeness Theorem, we know that there is a full model $\mathcal{F}_M$ and a valuation ν so that for every term N, $[\![N]\!]^{\nu}_{\mathcal{F}_M} = [\![M]\!]^{\nu}_{\mathcal{F}_M}$ iff $N =_{\beta\eta} M$. It then follows that M is in $\mathcal{L}_{IO}(\mathcal{G})$ iff $[\![M]\!]^{\nu}_{\mathcal{F}_M}$ is in $[\![\mathcal{L}_{IO}(\mathcal{G})]\!]^{\nu}_{\mathcal{F}_M}$. Since from Theorem 7 the set $[\![\mathcal{L}_{IO}(\mathcal{G})]\!]^{\nu}_{\mathcal{F}_M}$ can effectively be computed, and since the constructions of $\mathcal{F}_M$ and ν are also effective, it follows that we can decide whether $[\![M]\!]^{\nu}_{\mathcal{F}_M}$ is in $[\![\mathcal{L}_{IO}(\mathcal{G})]\!]^{\nu}_{\mathcal{F}_M}$. □

Theorem 8. *For every type α, there is a closed $\lambda Y+$-term M of type α so that $\mathcal{L}_{OI}(M)$ is the set of closed normal λ-terms of type α.*

Proof. Let us fix α, we are going to build a grammar $\mathcal{G}_\alpha$ so that $\mathcal{L}_{OI}(\mathcal{G}_\alpha)$ is the set of closed λ-terms of type α. For this we let $\mathcal{T}_\alpha$ be the *finite* set of types which are subformulae of α, *i.e.* the types that have a syntactic occurrence within α. We assume a total order on $\mathcal{T}_\alpha$ so that when we deal with a subset $S = \{\alpha_1, \ldots, \alpha_n\}$ of $\mathcal{T}_\alpha$, we take the ordering of the elements to be $\alpha_1, \ldots, \alpha_n$. We define the non-terminals of $\mathcal{G}_\alpha$ to be either pairs $\langle\gamma, S\rangle$ where γ is in $\mathcal{T}_\alpha$ and $S \subseteq \mathcal{T}_\alpha$ or to be of the form $\mathbf{cons}_\gamma$. The finiteness of $\mathcal{T}_\alpha$ guaranties that the set of non-terminals of $\mathcal{G}_\alpha$ is finite. The type of the non-terminals $\langle\gamma, S\rangle$ is $\alpha_1 \to \cdots \to \alpha_n \to \gamma$ when $S = \{\alpha_1, \ldots, \alpha_n\}$ (it is γ when $S = \emptyset$; and otherwise the order of the α_i is fixed according to the order of $\mathcal{T}_\alpha$), and the type of $\mathbf{cons}_\gamma$ is $\gamma \to \gamma \to \gamma$. The non-terminals $\mathbf{cons}_\gamma$ will serve to construct non-empty finite sets of variables of type γ; they allow to add a variable to such a set. For example the set of variables $\{x^\gamma, y^\gamma, z^\gamma\}$ will be represented as $\mathbf{cons}_\gamma x^\gamma(\mathbf{cons}_\gamma y^\gamma z^\gamma)$. The grammar will reduce such a term non-deterministically to one of the variables in the set

it represents. The non-terminals of the form $\langle \gamma, S\rangle$, where $S = \{\alpha_1, \ldots, \alpha_n\}$ will always be applied to terms representing non-empty sets of variables of type α_1, $\ldots$, α_n. If T_{V_1}, $\ldots$, T_{V_n} are terms representing respectively the finite set V_1 of variables of type α_1, $\ldots$, and the finite set V_n of variables of type α_n, the grammar will rewrite the term $\langle \gamma, S\rangle T_{V_1} \ldots T_{V_n}$ to any term of type γ and whose free variables are in the set $V_1 \cup \cdots \cup V_n$. In particular closed terms of type γ will be generated by the non-terminal $\langle \gamma, \emptyset\rangle$.

Let us now describe the rules of $\mathcal{G}_\alpha$. For the non-terminals $\langle \gamma, S\rangle$, let's assume that $S = \{\alpha_1, \ldots, \alpha_n\}$, then the rules of $\mathcal{G}_\alpha$ are given by:

1. $\mathbf{cons}_\gamma \to \lambda xy.x$ and $\mathbf{cons}_\gamma \to \lambda xy.y$ are rules of $\mathcal{G}_\alpha$. These two rules implement the non-deterministic choice of the grammar concerning a set of variables
2. in case $\gamma = \gamma_1 \to \gamma_2$ and $\gamma_1 = \alpha_i$ for some i in n, then

$$\langle \gamma, S\rangle \to \lambda y_1^{\alpha_1} \ldots y_n^{\alpha_n} x^{\alpha_i}.\langle \gamma_2, S\rangle y_1 \ldots y_{i-1}(\mathbf{cons}_{\alpha_i}\, x\, y_i) \ldots y_n$$

   is a rule of $\mathcal{G}_\alpha$. This rule constructs a λ-abstraction that introduces a fresh variable x; the variable x is added to the set of variables of type α_i; while the non-terminal $\langle \gamma_2, S\rangle$ is used to construct a term of type γ_2 with the updated sets of variables.
3. in case $\gamma = \gamma_1 \to \gamma_2$, $\gamma_1 \notin S$ and γ_1 appears in between α_i and α_{i+1} in the order of $\mathcal{T}_\alpha$, then

$$\langle \gamma, S\rangle \to \lambda y_1^{\alpha_1} \ldots y_n^{\alpha_n} x^{\gamma_1}.\langle \gamma_2, S \cup \{\gamma_1\}\rangle y_1 \ldots y_{i-1} x y_i \ldots y_n$$

   is a rule of $\mathcal{G}_\alpha$. This rule is similar to the previous one, except that γ_1 is not in S so that we initiate a new set of variables of type γ_1.
4. in case $\gamma = 0$ and $S \neq \emptyset$, if $\alpha_i = \beta_1 \to \cdots \to \beta_p \to 0$ then

$$\langle \gamma, S\rangle \to \lambda y_1^{\alpha_1} \ldots y_n^{\alpha_n}.y_i(\langle \beta_1, S\rangle y_1 \ldots y_n) \ldots (\langle \beta_p, S\rangle y_1 \ldots y_n)$$

   is a rule of $\mathcal{G}_\alpha$. This rule chooses one of the variable in the set of variables of type α_i represented by the variable y_i, and then inductively constructs the arguments of the right types for that variable.

It is rather easy to show that the non-terminals behave as we explained above. As a consequence the non-terminal $\langle \alpha, \emptyset\rangle$ generates all the closed terms of type α. □

B Proof of section 4

Lemma 1. *Given a $\lambda{+}\Omega$-term M^α, a monotone model $\mathcal{M} = (\mathcal{M}_\alpha)_{\alpha \in type}$, a hereditary prime valuation ν and a hereditary prime element f of $\mathcal{M}_\alpha$, we have the equivalence:*

$$f \leq [\![M^\alpha]\!]^\nu_{\mathcal{M}} \Leftrightarrow \exists N \in \mathcal{L}_{OI}(M).f \leq [\![N]\!]^\nu_{\mathcal{M}}$$

Proof. The direction from right to left is obvious and we are only going to prove the other direction. We assume that M^α is in β-normal and η-long form. We then proceed by induction on the structure of M.

In case $M = hM_1^{\gamma_1} \dots M_n^{\gamma_n}$, we have $\alpha = 0$, and since f is strictly greater than $\bot$, and $f \leq [\![M]\!]^\nu_{\mathcal{M}}$ we cannot have $h = \Omega^\gamma$. Thus h must be a variable x^γ variable with $\gamma = \gamma_1 \to \dots \to \gamma_n \to 0$. Since ν is hereditary prime there is $G \subseteq \mathcal{M}^+_\gamma$ so that $\nu(x) = \bigvee G$. Since $f \leq [\![M]\!]^\nu_{\mathcal{M}}$ and f is prime, there must be g in G so that $f \leq g[\![M_1]\!]^\nu_{\mathcal{M}} \dots [\![M_n]\!]^\nu_{\mathcal{M}} = g'$. But as g is in $\mathcal{M}^-_\gamma$, $g = g_1 \mapsto \dots \mapsto g_n \mapsto g'$ with $g_1, \dots, g_n$ respectively in $\mathcal{M}^+_{\gamma_1}, \dots, \mathcal{M}^+_{\gamma_n}$. Thus, by induction, for every i in $[1; n]$, there is N_i in $\mathcal{L}_{OI}(M_i)$ so that $g_i \leq [\![N_i]\!]^\nu_{\mathcal{M}}$. We then have that $N = xN_1 \dots N_n$ is in $\mathcal{L}_{OI}(M)$ moreover $g' \leq [\![N]\!]^\nu_{\mathcal{M}}$ so that $f \leq [\![N]\!]^\nu_{\mathcal{M}}$.

In case $M = \lambda x^{\alpha_1}.P^{\alpha_2}$, we have $f = f_1 \mapsto f_2$, where $f_1 = \bigvee F$ and $F \subseteq \mathcal{M}^-_{\alpha_1}$ and f_2 is in $\mathcal{M}^+_{\alpha_2}$. As $f \leq [\![M]\!]^\nu_{\mathcal{M}}$, we have that $f_2 \leq [\![P]\!]^{\nu[x^{\alpha_1} \leftarrow f_1]}_{\mathcal{M}}$. By induction hypothesis, there is Q in $\mathcal{L}_{OI}(P)$ so that $f_2 \leq [\![Q]\!]^{\nu[x_1^\alpha \leftarrow f_1]}_{\mathcal{M}}$. Thus, letting $N = \lambda x^{\alpha_1}.Q$, we have that $f \leq [\![N]\!]^\nu_{\mathbb{D}}$ and N in $\mathcal{L}_{OI}(M)$.

In case $M = M_1 + M_2$, we have $[\![M]\!]^\nu_{\mathcal{M}} = [\![M_1]\!]^\nu_{\mathcal{M}} \vee [\![M_2]\!]^\nu_{\mathcal{M}}$. Since f is in $\mathcal{M}^+_0$, f is prime and thus either $f \leq [\![M_1]\!]^\nu_{\mathcal{M}}$ or $f \leq [\![M_2]\!]^\nu_{\mathcal{M}}$. Let us assume, without loss of generality, that $f \leq [\![M_1]\!]^\nu_{\mathcal{M}}$. By induction hypothesis, there is N in $\mathcal{L}_{OI}(M_1)$ so that $f \leq [\![N]\!]^\nu_{\mathcal{M}}$. The conclusion follows from the fact that N is an element of $\mathcal{L}_{OI}(M)$. □

We here start with the proof of Theorem 13. For this we first need to introduce the notion of logical relation.

Given two monotone models $\mathcal{M} = (\mathcal{M}_\alpha)_{\alpha \in type}$ and $\mathcal{N} = (\mathcal{N}_\alpha)_{\alpha \in type}$ a logical relation $\mathcal{R}$ between $\mathcal{M}$ and $\mathcal{N}$ is a family $(\mathcal{R}_\alpha)_{\alpha \in type}$ so that $\mathcal{R}_\alpha \subseteq \mathcal{M}_\alpha \times \mathcal{N}_\alpha$ and verifying that $\mathcal{R}_{\alpha\to\beta} = \{(f, g) \in \mathcal{M}_{\alpha\to\beta} \times \mathcal{N}_{\alpha\to\beta} \mid \forall (f', g') \in \mathcal{R}_\alpha.(f(f'), g(g')) \in \mathcal{R}_\beta\}$. Notice that a logical relation $\mathcal{R}$ is completely determined by $\mathcal{R}_0$. In general, when the type is irrelevant we write $f\ R\ g$ to mean that $(f, g) \in \mathcal{R}_\alpha$ for some appropriate α. Given two valuations ν and μ, we write $\nu\ \mathcal{R}\ \mu$ when for all variable x, $\nu(x)\ \mathcal{R}\ \mu(x)$. Logical relations satisfy the following property:

Lemma 3 (Fundamental Lemma). *Given two monotone models $\mathcal{M}$ and $\mathcal{N}$, a logical relation $\mathcal{R}$ between $\mathcal{M}$ and $\mathcal{N}$, and two valuations ν and μ respectively over $\mathcal{M}$ and $\mathcal{N}$, so that $\nu\ \mathcal{R}\ \mu$, then for every $\lambda Y{+}\Omega$-term M we have $[\![M]\!]^\nu_{\mathcal{M}}\ \mathcal{R}\ [\![M]\!]^\mu_{\mathcal{N}}$.*

We now turn to the proof of Theorem 13 itself. This requires some terminology and a certain number of technical lemmas.

Given A and B two finite sets such that $A \subseteq B$, we define $\mathcal{I}_{A,B} = (\mathcal{I}_{A,B,\alpha})_{\alpha \in type}$ for the logical relation between $\mathcal{M}(A)$ and $\mathcal{M}(B)$ such that $\mathcal{I}_{A,B,0} = \{(C \cap A, C) \mid C \subseteq B\}$. We now define a function $E_{A,B}$ that maps every element f of $\mathcal{M}_\alpha(A)$ to an element $E_{A,B}(f)$ of $\mathcal{M}_\alpha(B)$, and such that:

- $E_{A,B}(f) = f$ when $\alpha = 0$,
- $E_{A,B}(f) = \bigvee\{E_{A,B}(g) \mapsto E_{A,B}(f(g)) \mid g \in \mathcal{M}_\beta(A)\}$ when $\alpha = \beta \to \gamma$.

We are now showing some basic properties of $\mathcal{I}$ and $E_{A,B}$.

Lemma 4. *If (f_1, g_1) and (f_2, g_2) are in $\mathcal{I}_{A,B,\alpha}$ then $(f_1 \vee f_2, g_1 \vee g_2)$ is in $\mathcal{I}_{A,B,\alpha}$.*

Proof. The proof is done by induction on the structure of α. In case $\alpha = 0$, we have that $f_1 = g_1 \cap A$ and $f_2 = g_2 \cap A$ then $f_1 \vee f_2 = f_1 \cup f_2 = (g_1 \cap A) \cup (g_2 \cap A) = (g_1 \cup g_2) \cap A = (g_1 \vee g_2) \cap A$ and thus $(f_1 \vee f_2, g_1 \vee g_2)$ is in $\mathcal{I}_{A,B,\alpha}$. In case $\alpha = \beta \to \gamma$, then given (h, l) in $\mathcal{I}_{A,B,\beta}$ we have that $(f_1(h), g_1(l))$ and $(f_2(h), g_2(l))$ are in $\mathcal{I}_{A,B,\gamma}$ so that by induction $(f_1(h) \vee f_2(h), g_1(l) \vee g_2(l)) = (f_1 \vee f_2(h), g_1 \vee g_2(l))$ is in $\mathcal{I}_{A,B,\gamma}$ showing finally that $(f_1 \vee f_2, g_1 \vee g_2)$ is in $\mathcal{I}_{A,B,\alpha}$. □

Lemma 5. *For every type α, $(\bot_{A,\alpha}, \bot_{B,\alpha})$ is in $\mathcal{I}_{A,B,\alpha}$.*

Proof. A simple induction on α. □

Lemma 6. *If $(\bot_{A,\alpha}, f)$ is in $\mathcal{I}_{A,B,\alpha}$ and $(\bot_{A,\beta}, g)$ is in $\mathcal{I}_{A,B,\beta}$ then $(\bot_{A,\alpha\to\beta}, f \mapsto g)$ is in $\mathcal{I}_{A,B,\alpha\to\beta}$.*

Proof. Given (h, l) is in $\mathcal{I}_{A,B,\alpha}$, we have $f \mapsto g(l) = g$ or $f \mapsto g(l) = \bot_{B,\beta}$ and in each case $\bot_{A,\alpha\to\beta}(h) = \bot_{A,\beta}$. As we have, by assumption $(\bot_{A,\beta}, g)$ in $\mathcal{I}_{A,B,\beta}$ and, by Lemma 4, $(\bot_{A,\beta}, \bot_{B,\beta})$ also in $\mathcal{I}_{A,B,\beta}$ which gives that $(\bot_{A,\alpha\to\beta}, f \mapsto g)$ is in $\mathcal{I}_{A,B,\alpha\to\beta}$. □

The next lemma is central in the proof of Statman Theorem, it allows us to use disjoint parts of a model so that they don't interfere.

Lemma 7. *If A_1 and A_2 are disjoint subsets of B, and f is in $\mathcal{M}_\alpha(A_1)$ then we have:*

1. *$(\bot_{A_2,\alpha}, E_{A_1,B}(f))$ is in $\mathcal{I}_{A_2,B,\alpha}$*
2. *if $\alpha = \beta \to \gamma$ and g is in $\mathcal{M}_\beta(B)$ then $(\bot_{A_2,\gamma}, E_{A_1,B}(f)(g))$ is in $\mathcal{I}_{A_2,B,\gamma}$.*

Proof. 1. We proceed by induction on α. In case $\alpha = 0$, $\bot_{A_2,\alpha} = \emptyset$ and since $f \subseteq A_1$ and A_1 is disjoint from A_2 we have that $f \cap A_2 = \emptyset$. It thus follows that $(\emptyset, f)$ is well in $\mathcal{I}_{A_2,B,\alpha}$. In case $\alpha = \beta \to \gamma$, let (h, l) be in $\mathcal{I}_{A_2,B,\beta}$, we need to show that $(\bot_{A_2,\gamma}, E_{A_1,B}(f)(l))$ is in $\mathcal{I}_{A_2,B,\gamma}$. We have that $E_{A_1,B}(f)(l) = \bigvee\{E_{A_1,B}(f(k)) \mid E_{A_1,B}(k) \leq l\}$. As, by induction, we have that $(\bot_{A_2,\gamma}, E_{A_1,B}(f(k)))$ is in $\mathcal{I}_{A_2,B,\gamma}$ and that, by the previous Lemma $(\bot_{A_2,\gamma}, \bot_{B,\gamma})$ is in $\mathcal{I}_{A_2,B,\gamma}$, we obtain, using iteratively Lemma 4, that $(\bot_{A_2,\gamma}, E_{A_1,B}(f)(l))$ is in $\mathcal{I}_{A_2,B,\gamma}$.
2. by definition of $E_{A_1,B}(f)$, we have $E_{A_1,B}(f)(g) = \bigvee\{E_{A_1,B}(f(h)) \mid E_{A_1,B}(h) \leq g\}$. From the previous statement of the Lemma, we have $(\bot_{A_2,\gamma}, E_{A_1,B}(f(h)))$ in $\mathcal{I}_{A_2,B,\gamma}$; Lemma 4 gives that $(\bot_{A_2,\gamma}, \bigvee\{E_{A_1,B}(f(h)) \mid E_{A_1,B}(h) \leq g\})$ is in $\mathcal{I}_{A_2,B,\gamma}$ which gives the result. □

Lemma 8. *Given a type α we have the following properties:*

1. *for every f in $\mathcal{M}_\alpha(A)$, $(f, E_{A,B}(f))$ is in $\mathcal{I}_{A,B,\alpha}$,*
2. *for every (f, g) in $\mathcal{I}_{A,B,\alpha}$, for every h in $\mathcal{M}_\alpha(A)$, we have $E_{A,B}(h) \leq g$ iff $h \leq f$.*

Proof. We proceed by induction on the structure of α. The case where $\alpha = 0$ is clear.

In case $\alpha = \beta \to \gamma$, we have:

1. given (g_1, g_2) in $\mathcal{I}_{A,B,\beta}$, we want to show that $(f(g_1), E_{A,B}(f)(g_2))$ is in $\mathcal{I}_{A,B,\gamma}$. By definition, $E_{A,B}(f)(g_2) = \bigvee\{E_{A,B}(f(l)) \mid E_{A,B}(l) \leq g_2\}$. But by induction hypothesis, given l in $\mathcal{M}_\beta(A)$, $E_{A,B}(l) \leq g_2$ iff $l \leq g_1$. Therefore $E_{A,B}(f)(g_2) = \bigvee\{E_{A,B}(f(l)) \mid l \leq g_1\}$. But the induction hypothesis also implies that $(f(g_1), E_{A,B}(f(g_1)))$ is in $\mathcal{I}_{A,B,\gamma}$, and by monotonicity of f, if $l \leq g_1$, then $f(l) \leq f(g_1)$, which by induction is equivalent to $E_{A,B}(f(l)) \leq E_{A,B}(f(g_1))$. Therefore $\bigvee\{E_{A,B}(f(l)) \mid l \leq g_1\} = E_{A,B}(f(g_1))$, $E_{A,B}(f)(g_2) = E_{A,B}(f(g_1))$ and $(f(g_1), E_{A,B}(f)(g_2))$ is in $\mathcal{I}_{A,B,\gamma}$.
2. Let's first prove that if $h \leq f$ then $E_{A,B}(h) \leq g$. As $h \leq f$, we have we have that for l in $\mathcal{M}_\beta(A)$, $h(l) \leq f(l)$ and by induction we have $(f(l), E_{A,B}(f(l)))$ in $\mathcal{I}_{A,B,\gamma}$ so that, by induction again, $E_{A,B}(h(l)) \leq E_{A,B}(f(l))$. But, given k in $\mathcal{M}_\beta(B)$, we have

$$E_{A,B}(h)(k) = \bigvee\{E_{A,B}(h(l)) \mid E_{A,B}(l) \leq k\}$$
$$E_{A,B}(f)(k) = \bigvee\{E_{A,B}(f(l)) \mid E_{A,B}(l) \leq k\}$$

and since we have seen that for every l in $\mathcal{M}_\beta(A)$, $E_{A,B}(h(l)) \leq E_{A,B}(f(l))$, we necessarily have $E_{A,B}(h)(k) \leq E_{A,B}(f)(k)$ and therefore $E_{A,B}(h) \leq E_{A,B}(f)$. But by induction we have that for every l in $\mathcal{M}_\beta(A)$, $(l, E_{A,B}(l))$ is in $\mathcal{I}_{A,B,\beta}$ thus $(f(l), g(E_{A,B}(l)))$ is in $\mathcal{I}_{A,B,\gamma}$. By induction, we have $E_{A,B}(f(l)) \leq g(E_{A,B}(l))$. Now given k in $\mathcal{M}_\beta(A \cup B)$, we have that $E_{A,B}(f)(k) = \bigvee\{E_{A,B}(f(l)) \mid E_{A,B}(l) \leq k\}$, but as when $E_{A,B}(l) \leq k$, $g(E_{A,B}(l)) \leq g(k)$, we have $E_{A,B}(f(l)) \leq g(k)$ and thus $E_{A,B}(f)(k) \leq g(k)$ proving $E_{A,B}(f) \leq g$ and since we already showed $E_{A,B}(h) \leq E_{A,B}(f)$ we have $E_{A,B}(h) \leq g$.
Let's now suppose that $E_{A,B}(h) \leq g$, given l in $\mathcal{M}_\beta(A)$, by induction $(l, E_{A,B}(l))$ is in $\mathcal{M}_\beta(A)$. Therefore, $(f(l), g(E_{A,B}(l)))$ is in $\mathcal{I}_{A,B,\gamma}$, but also, $E_{A,B}(h)(E_{A,B}(l)) \leq g(E_{A,B}(l))$. As $E_{A,B}(h)(E_{A,B}(l)) = \bigvee\{E_{A,B}(h(l')) \mid E_{A,B}(l') \leq E_{A,B}(l)\}$ and that by induction $E_{A,B}(l') \leq E_{A,B}(l)$ iff $l' \leq l$, we have $E_{A,B}(h)(E_{A,B}(l)) = E_{A,B}(h(l))$. Thus, from $E_{A,B}(h)(E_{A,B}(l)) \leq g(E_{A,B}(l))$ we obtain that $E_{A,B}(h(l)) \leq g(E_{A,B}(l))$, then the induction hypothesis gives that $h(l) \leq f(l)$ and finally that $h \leq f$. □

Lemma 9. *Given (f, g) in $\mathcal{I}_{A,B,\beta}$ and h in $\mathcal{M}_\alpha(A)$, we have*

$$(h \mapsto f, E_{A,B}(h) \mapsto g) \text{ is in } \mathcal{I}_{A,B,\alpha\to\beta}.$$

Proof. Given (k, l) in $\mathcal{I}_{A,B,\alpha}$, from Lemma 8 we have that $h \leq k$ iff $E_{A,B}(h) \leq l$. Therefore we have that either $(h \mapsto f(k), E_{A,B}(h) \mapsto g(l)) = (f, g)$ or $(h \mapsto f(k), E_{A,B}(h) \mapsto g(l)) = (\bot_{A,\beta}, \bot_{B,\beta})$ and in both case we have that $(h \mapsto f(k), E_{A,B}(h) \mapsto g(l))$ is in $\mathcal{I}_{A,B,\beta}$. □

Theorem 13. *For every type α and every pure term M, one can effectively construct a triple T that is characteristic of M.*

Proof. We assume that M is in η-long form and we proceed by induction on M. There are two cases.

Case $M = x^\alpha M_1 \dots M_n$. Let us assume that $\alpha = \alpha_1 \to \dots \to \alpha_n \to 0$; by induction hypothesis, for every i in $[1,n]$ there is $T_i = (A_i, \nu_i, f_i)$ that is characteristic of M_i. We let $B = \{b\} \cup \bigcup_{i=1}^n A_i$ where b is not in any of the A_i, we let $A_{n+1} = \{b\}$, and we also let ν be the valuation so that, for every variable y^β different from x^α:

$$\nu(y^\beta) = \bigvee_{i=1}^n E_{A_i,B}(\nu_i(y^\beta))$$

and

$$\nu(x^\alpha) = E_{A_1,B}(f_1) \mapsto \dots \mapsto E_{A_n,B}(f_n) \mapsto \{b\} \vee \bigvee_{i=1}^n E_{A_i,B}(\nu_i(x^\alpha))$$

We are going to see that $T = (B, \nu, \{b\})$ is a characteristic triple of M. It is easy to see that $\{b\}$ is hereditary prime and that ν is a hereditary prime valuation. Thus to prove that T is a characteristic triple of M, it just remains to prove that for every λ-term N, $M =_{\beta\eta} N$ iff $\{b\} \leq [\![N]\!]^\nu_{\mathcal{M}(B)}$.

Lemmas 7 gives that, for every variable y^β, every (i,j) in $[1,n+1] \times [1,n]$, if $i \neq j$ then $(\bot_{A_i,\beta}, E_{A_j,B}(\nu_j(y^\beta)))$ is in $\mathcal{I}_{A_i,B,\beta}$. An iterative use of Lemma 4 then shows that

$$(\bot_{A_i,\beta}, \bigvee_{j \in \{1,\dots,n\}-\{i\}} E_{A_j,B}(\nu_j(y^\beta)) \in \mathcal{I}_{A_i,B,\beta}$$

Notice that instantiating in this statement i with $n+1$ gives

$$(\bot_{A_{n+1},\beta}, \bigvee_{j \in \{1,\dots,n\}} E_{A_j,B}(\nu_j(y^\beta)) \in \mathcal{I}_{A_i,B,\beta}$$

Moreover Lemma 8 implies that $(\nu_i(y^\beta), E_{A_i,B}(\nu_i(y^\beta)))$ is in $\mathcal{I}_{A_i,B,\beta}$. Another use of Lemma 4 then gives that for every i in $[1,n]$, $(\nu_i(y^\beta), \bigvee_{j\in[1,n]} E_{A_j,B}(\nu_j(y^\beta)))$ is in $\mathcal{I}_{A_i,B,\beta}$. So that when $y^\beta \neq x^\alpha$, $(\nu_i(y^\beta), \nu(y^\beta))$ is in $\mathcal{I}_{A_i,B,\beta}$ for each i in $[1,n]$ and $(\bot_{A_{n+1},\beta}, \nu(y^\beta))$ is in $\mathcal{I}_{A_i,B,\beta}$. We are now going to see that we also have $(\nu_i(x^\alpha), \nu(x^\alpha))$ is in $\mathcal{I}_{A_i,B,\alpha}$ for each i in $[1,n]$. To prove this, let $g = E_{A_1,B}(f_1) \mapsto \dots \mapsto E_{A_n,B}(f_n) \mapsto \{b\}$ and let us prove that for every i in $[1,n]$, $(\bot_{A_i,\alpha}, g)$ is in $\mathcal{I}_{A_i,B,\alpha}$. Let $g_0 = \{b\}$, $\delta_0 = 0$ and $g_{i+1} = E_{A_{n-i},B}(f_{n-i}) \mapsto g_i$ and $\delta_{i+1} = \alpha_{n-i} \to \delta_i$. Notice that $g_{n-1} = g$ and $\delta_{n-1} = \alpha$. We prove by induction on i that for every i in $[0,n-1]$ and every j in $[1,n]$, $(\bot_{A_j,\delta_i}, g_i)$ is in $\mathcal{I}_{A_j,B,\delta_i}$. The case where $i = 0$ is clear. Let's now show that if $(\bot_{A_j,\delta_i}, g_i)$ is in $\mathcal{I}_{A_j,B,\delta_i}$ then $(\bot_{A_j,\delta_{i+1}}, g_{i+1})$ is in $\mathcal{I}_{A_j,B,\delta_{i+1}}$. There are two cases depending on whether $i = j$ or not. If $i = j$, then, Lemma 9, gives that $(f_{n-i} \mapsto \bot_{A_i,\delta_i}, E_{A_{n-i},B}(f_{n-i}) \mapsto g_i) = (\bot_{A_i,\delta_{i+1}}, g_{i+1})$ is in $\mathcal{I}_{A_i,B,\delta_{i+1}}$. If $i \neq j$, then Lemma 7 gives that $(\bot_{A_j,\alpha_{n-i}}, E_{A_i,B}(f_{n-i}))$ is in $\mathcal{I}_{A_j,B,\alpha_{n-i}}$ and Lemma 6

thus gives that $(\bot_{A_i,\delta_{i+1}}, g_{i+1})$ is in $\mathcal{I}_{A_i,B,\delta_{i+1}}$. Thus finally we have that, for every i, $(\bot_{A_i,\alpha}, g)$ is in $\mathcal{I}_{A_i,B,\alpha}$ and thus $(\nu_i(x^\alpha), \nu(x^\alpha))$ is in $\mathcal{I}_{A_i,B,\alpha}$.

Using the fundamental Lemma of logical relations, given a λ-term N_i of type α_i, we obtain that

$$([\![N_i]\!]^{\nu_i}_{\mathcal{M}(A_i)}, [\![N_i]\!]^{\nu}_{\mathcal{M}(B)}) \in \mathcal{I}_{A_i,B,\alpha_i}$$

moreover, from Lemma 8, we have that $E_{A_i,B}(f_i) \leq [\![N_i]\!]^{\nu}_{\mathcal{M}(B)}$ iff $f_i \leq [\![N_i]\!]^{\nu_i}_{\mathcal{M}(A_i)}$, but, since T_i is a characteristic triple for M_i, we obtain that $E_{A_i,B}(f_i) \leq [\![N_i]\!]^{\nu}_{\mathcal{M}(B)}$ iff $N_i =_\beta M_i$.

Let us now suppose that we are given a λ-term N of type 0 such that $\{b\} \leq [\![N]\!]^{\nu}_{\mathcal{M}(B)}$. Without loss of generality, we assume that N is in β-normal, η-long form. We then prove that $N =_\beta M$. We must have $N = y^\beta N_1 \dots N_p$. If y^β is different from x^α, then, we have by Lemma 7 that $(\bot_{A_{n+1},B,\beta}, E_{A_i,B,\beta}(\nu_i(y^\beta)))$ is in $\mathcal{I}_{A_{n+1},B,\beta}$. Therefore, by Lemma 7 again, we obtain that

$$(\bot_{A_{n+1},0}, E_{A_i,B,\beta}(\nu_i(y_\beta))[\![N_1]\!]^{\nu}_{\mathcal{M}(B)} \dots [\![N_n]\!]^{\nu}_{\mathcal{M}(B)}) \in \mathcal{I}_{A_{n+1},B,0}$$

so that by Lemma 4 $(\bot_{A_{n+1},0}, [\![N]\!]^{\nu}_{\mathcal{M}})$ is also in $\mathcal{I}_{A_{n+1},B,0}$ which is possible only if $b \notin [\![N]\!]^{v}_{\mathcal{M}(B)}$ or, equivalently, only if we do not have $\{b\} \leq [\![N]\!]^{\nu}_{\mathcal{M}(B)}$. Therefore if $\{b\} \leq [\![N]\!]^{v}_{\mathcal{M}(B)}$ we must have $y^\beta = x^\alpha$ and $N = x^\alpha N_1 \dots N_n$. We are now going to show that we must also have $E_{A_i,B}(f_i) \leq [\![N_i]\!]^{v}_{\mathcal{M}(B)}$, which, as we have seen above, is equivalent to $N_i =_\beta M_i$ and thus to $N =_\beta M$. Let's suppose that for some i we do not have $E_{A_i,B}(f_i) \leq [\![N_i]\!]^{v}_{\mathcal{M}(B)}$, this means that $g([\![N_1]\!]^{v}_{\mathcal{M}(B)}) \dots ([\![N_n]\!]^{v}_{\mathcal{M}(B)}) = \bot_{B,0}$ and that

$$\nu(x^\alpha)([\![N_1]\!]^{\nu}_{\mathcal{M}(B)}) \dots ([\![N_n]\!]^{\nu}_{\mathcal{M}(B)}) = \bigvee_{i\in[1,n]} E_{A_i,B}(\nu_i(x^\alpha))([\![N_1]\!]^{\nu}_{\mathcal{M}(B)}) \dots ([\![N_n]\!]^{\nu}_{\mathcal{M}(B)})$$

but then, we have that $(\bot_{A_{n+1},B,\alpha}, E_{A_i,B}(\nu_i(x^\alpha)))$ is in $\mathcal{I}_{A_{n+1},B,\alpha}$. This implies that for all i $(\bot_{A_{n+1},B,0}, E_{A_i,B}(\nu_i(x^\alpha))([\![N_1]\!]^{\nu}_{\mathcal{M}(B)}) \dots ([\![N_n]\!]^{\nu}_{\mathcal{M}(B)}))$ is in $\mathcal{I}_{A_{n+1},B,0}$ and, as we did above, Lemma 7 gives that it cannot be the case that $\{b\} \leq \nu(x^\alpha)([\![N_1]\!]^{v}_{\mathcal{M}(B)}) \dots ([\![N_n]\!]^{v}_{\mathcal{M}(B)})$. Therefore, for every i in $[1, n]$ we must have $E_{A_i,B}(f_i) \leq [\![N_i]\!]^{v}_{\mathcal{M}(B)}$.

Case $M = \lambda x^\beta.P$. We assume that P is of type γ and $\alpha = \beta \to \gamma$. By induction there is a triple $U = (A, \nu, f)$ that is characteristic of P; we let $T = (A, \nu', g)$ with $\nu' = \nu[x^\alpha \leftarrow \bot_{A,\alpha}]$ and $g = \nu(x^\alpha) \mapsto f$. Clearly, g is hereditary prime and ν' is a hereditary prime valuation.

To prove that T is a characteristic triple of M, it just remains to show that for every λ-term N, $g \leq [\![N]\!]^{\nu'}_{\mathcal{M}(A)}$ iff $N =_\beta M$. So, given N such that $g \leq [\![N]\!]^{\nu'}_{\mathcal{M}(A)}$; we have that $g(\nu(x^\alpha)) \leq [\![N]\!]^{\nu'}_{\mathcal{M}(A)}(\nu(x^\alpha))$ so that $f \leq [\![Nx^\alpha]\!]^{\nu}_{\mathcal{M}(A)}$, therefore $Nx^\alpha =_\beta P$ and thus $\lambda x^\alpha.Nx^\alpha =_\beta \lambda x^\alpha.P$ finally giving $N =_\beta M$. □