FREC delivrable 4: Recognizability for λ-terms

This document constitutes the delivrable 4 of the ANR project FREC. It describes the foundations of recognizability in the simply typed λ-calculus. It is divided in two parts:

1. the first part describes the notion, shows how it relates to the notions of recognizability in free algebras and in free monoids. It also gives some examples of applications in the setting of λ-calculus by providing elegant solutions to difficult problems.
2. the second part (which was written in collaboration with Giulio Manzonetto, Mai Gehrke, and Henk Barendregt) describes in detail the correspondence between the "algebraic view" and the "automata theoretic view" of recognizability in simply typed λ-calculus. This correspondence is glossed over in the first part. It shows that intersection types, representing automata in the setting of λ-calculus, and finite models, representing algebra in this setting, have the same expressive power for defining sets of λ-term. This part is also more oriented towards connecting two problems of λ-calculus: the inhabitation problem for intersection types and the λ-definability problem.

Part I

Recognizability in the λ-calculus

1 Introduction

Formal language theory is mainly concerned with the study of structures like strings, trees or even graphs. In this paper we try to add simply typed λ-terms to the scope of this theory. This article is a first step: the definition of recognizable sets which are a fundamental notion of formal language theory.

Languages of λ-terms appear in several research areas, but there has been really few research explicitly mentioning them and even fewer studying them. To our knowledge the first work explicitly defining a notion of language of λ-terms is that of de Groote [1]. In mathematical linguistics, the pioneering work Montague [2] shows how to connect syntax and semantics of natural language with the simply typed λ-calculus. Syntactic structures are interpreted via a homomorphism built with λ-terms. The normal forms obtained this way denote formulae of higher-order logic whose interpretation in a suitable model gives the semantics of the sentence. The set of formulae that this technique allows to generate can be seen as a language and one may wonder whether such a language can be *parsed* similarly to other languages like context-free languages. Parsing such languages results in generating sentences from their meaning representation. We have showed that this could effectively be done [3].

Still in mathematical linguistics, the type-logical tradition originating from Lambek's work [4], defines syntactic structures as proofs in some substructural logic. Several proposals have emerged in order to control the structure of those proofs such as in Moortgat's work [5] and his followers. These proofs may be represented as simply typed λ-terms and the set of syntactic structures defines a language of λ-terms.

Since simply typed λ-terms generalize both strings and trees, a notion of recognizable language of simply typed λ-terms should naturally extend those already defined for strings and trees. Furthermore, these languages should also be closed under the operational semantics of the λ-calculus, *i.e.* $\beta\eta$-convertibility. The easiest way to obtain such an extension is to use the algebraic characterization of recognizable sets of strings or trees which says that recognizable sets are precisely the sets that are the union of equivalence classes of a finite congruence. Generalizing this definition to sets of simply typed λ-terms consists in saying that such sets are recognizable if and only if they the set of terms that are interpreted as certain points in a finite model. But such a definition may not be useful in certain situations, this is the reason why we need a notion of automaton of λ-terms that coincides with that of recognizability. We define such automata using intersection types.

This work provides a natural framework in which several results that have appeared in the literature on simply typed λ-calculus can be related. In particular, our work shows that Urzyczyn's result on the undecidability of the emptiness problem for intersection types [6] can be seen as a corollary of Loader's result on the undecidability of λ-definability [7]. This shall be developed in greater details in the second part of the document. Moreover, we have showed in [3] that the singleton language can be defined with intersection types, the equivalence we establish here between recognizability in terms of finite models and in

terms of automata gives an alternate proof of Statman's completeness theorem [8] (see also [9]). Furthermore, Statman [8] has showed that higher order matching is related to λ-definability. Since our notion of recognizability is related to λ-definability it gives tools with which we can study this problem.

The paper is organized as follows: section 2 gives the necessary definitions, section 3 gives the definition of recognizable sets of λ-terms. In section 4 we give an automaton-like characterization of recognizability. Section 5 gives its closure properties and section 6 shows some basic applications of the notion of recognizability for parsing and higher order matching. Finally we conclude in section 7.

2 Preliminaries

We here briefly review various notions concerning the simply typed λ-calculus and some related notions.

2.1 Simply typed λ-calculus

Higher Order Signatures (HOS) declare a finite number of constants by assigning them types. An HOS Σ is a triple $(\mathcal{A}, \mathcal{C}, \tau)$, where $\mathcal{A}$ is a finite set of *atomic types*, from which the set of *complex types*, $\mathcal{T}_\Sigma$, is built using the binary infix operator $\rightarrow$, $\mathcal{C}$ is a finite set of constants and τ is a function from $\mathcal{C}$ to $\mathcal{T}_\Sigma$. As usual, we will consider that $\rightarrow$ associates to the right and write $\alpha_1 \rightarrow \cdots \rightarrow \alpha_n \rightarrow \alpha$ for the type $(\alpha_1 \rightarrow (\cdots \rightarrow (\alpha_n \rightarrow \alpha) \cdots))$. The *order* of a type α, $ord(\alpha)$, is 1 when α is atomic and $\max(ord(\alpha_1) + 1, ord(\alpha_2))$ when $\alpha = \alpha_1 \rightarrow \alpha_2$. By extension, the order of a HOS is the maximal order of type it associates to a constant. We suppose that we are given an infinite countable set of λ-variables V. We use types *à la Church* and variables explicitly carry their types. So we will write x^α, y^α or z^α (possibly with indices) the elements of $\mathcal{V} \times \{\alpha\}$, the variables of type α. A HOS Σ defines a set of simply typed λ-terms Λ_Σ. This set is the union of the sets of the family $(\Lambda^\alpha_\Sigma)_{\alpha \in \mathcal{T}_\Sigma}$ defined as the smallest sets verifying:

1. $x^\alpha \in \Lambda^\alpha_\Sigma$,
2. for $c \in \mathcal{C}$, $c \in \Lambda^{\tau(c)}_\Sigma$,
3. if $M_1 \in \Lambda^{\alpha \rightarrow \beta}_\Sigma$ and $M_2 \in \Lambda^\alpha_\Sigma$ then $(M_1 M_2) \in \Lambda^\beta_\Sigma$,
4. if $M \in \Lambda^\beta$ then $\lambda x^\alpha . M \in \Lambda^{\alpha \rightarrow \beta}_\Sigma$.

We take for granted the notions of free variables, closed terms, substitution, the various notions of conversions and reductions associated to the λ-calculus, the notions of normal form, of η-long form (or long form) and the notion of linearity. We write $\Lambda^\alpha_{\Sigma,W}$ to designates the set of terms of type α built on Σ whose set of free variables is included in W.

2.2 Trees and strings as λ-terms

A HOS is said to be a tree-HOS when it is a second order HOS and that it uses only one type namely o. We write $o^n \to o$ in the place of $\underbrace{o \to \cdots \to o}_{n} \to o$ and say that a constant of type $o^n \to o$ has *arity* n. It is easy to see that every freely and finitely generated sets of ranked trees can be seen as a the closed normal terms of type o built on a tree-HOS.

A HOS is said to be a string-HOS when it is a tree-HOS whose constants all have arity 1. Strings are represented as closed terms of type $o \to o$ and the string $c_1 \dots c_n$ is represented by the term $\lambda x^o.c_1(\dots(c_n x^o)\dots)$ denoted by $/c_1 \dots c_n/$. The empty string is $\lambda x^o.x^o$ and the concatenation operation can be represented by function composition $\lambda x^o.s_1(s_2\, x^o)$ (*c.f.* [1]).

2.3 Homomorphisms

A *homomorphism* between the signatures Σ_1 and Σ_2 is a pair (g, h) such that g maps $\mathcal{T}_{\Sigma_1}$ to $\mathcal{T}_{\Sigma_2}$, h maps Λ_{Σ_1} to Λ_{Σ_2} and verify the following properties:

1. $g(\alpha \to \beta) = g(\alpha) \to g(\beta)$,
2. $h(x^\alpha) = x^{g(\alpha)}$,
3. $h(c)$ is a closed term of $\Lambda^{g(\tau(c))}_{\Sigma_2}$,
4. $h(M_1 M_2) = h(M_1)h(M_2)$ and
5. $h(\lambda x^\beta.M) = \lambda x^{g(\beta)}.h(M)$.

A homomorphism is said to be *linear* whenever constants are mapped to linear terms. We write $\mathcal{H}(\alpha)$ and $\mathcal{H}(M)$ respectively instead of $g(\alpha)$ and of $h(M)$ for a given homomorphism $\mathcal{H} = (g, h)$. Note that if $\mathcal{H}$ is a homomorphism from Σ_1 to Σ_2 and $M \in \Lambda^\alpha_{\Sigma_1}$ then $\mathcal{H}(M) \in \Lambda^{\mathcal{H}(\alpha)}_{\Sigma_2}$.

The order of a homomorphism $\mathcal{H}$ is the maximal order of the type it associates to an atomic type. The usual notion of tree-homomorphism (*resp.* string homomorphism) is a first order homomorphism (in our sense) between tree-signatures (*resp.* string-signatures). A first order homomorphism between Σ_1 and Σ_2 that maps constants of Σ_1 to constants of Σ_2 is called a *relabeling*.

2.4 Models

Given a signature Σ, a *full model* of Σ is a pair $\mathbb{M} = ((\mathcal{M}^\alpha)_{\alpha \in \mathcal{T}_\Sigma}, \rho)$ where:

1. $(\mathcal{M}^\alpha)_{\alpha \in \mathcal{T}_\Sigma}$ is a family of sets verifying:
   (a) for all $\alpha, \beta \in \mathcal{T}_\Sigma$, $\mathcal{M}^{\alpha \to \beta} = \mathcal{M}^{\alpha \mathcal{M}^\beta}$,
   (b) the sets $\mathcal{M}^\alpha$ such that α is atomic are pairwise disjoint.
2. ρ is a function from $\mathcal{C}$ to $\mathcal{M}^\alpha$ so that $\alpha = \rho(c)$.

A full model, $\mathbb{M} = ((\mathcal{M}^\alpha)_{\alpha \in \mathcal{T}_\Sigma}, \rho)$, of Σ is said *finite* when for all $\alpha \in \mathcal{T}_\Sigma$, $\mathcal{M}^\alpha$ is a finite set. Remark that $\mathbb{M}$ is finite if and only if for all atomic types α, $\mathcal{M}^\alpha$ is finite.

Given $\mathbb{M} = ((\mathcal{M}^\alpha)_{\alpha\in\mathcal{T}_\Sigma}, \rho)$ a full model of Σ, the terms of Λ^α_Σ are interpreted as elements of $\mathcal{M}^\alpha$. This interpretation necessitates the definition of *variable assignments* which are partial functions that associate elements of $\mathcal{M}^\alpha$ to variables like x^α. A variable assignment is said *finite* when its domain is finite. Given a variable assignment ν, a variable x^α and $m \in \mathcal{M}^\alpha$ we define $\nu[x^\alpha \leftarrow m]$ to be the variable assignment verifying:

$$\nu[x^\alpha \leftarrow m](y^\beta) \begin{cases} m \text{ if } y^\beta = x^\alpha \\ \nu(y^\beta) \text{ otherwise} \end{cases}$$

Given a full model $\mathbb{M} = ((\mathcal{M}^\alpha)_{\alpha\in\mathcal{T}_\Sigma}, \rho)$ a variable assignment ν, the interpretation of the elements of Λ_Σ (whose sets of free variables are included in the domain of definition of ν) in $\mathbb{M}$ is inductively defined as follows:

1. $[\![x^\alpha]\!]^{\mathbb{M}}_\nu = \nu(x^\alpha)$
2. $[\![c]\!]^{\mathbb{M}}_\nu = \rho(c)$
3. $[\![M_1 M_2]\!]^{\mathbb{M}}_\nu = [\![M_1]\!]^{\mathbb{M}}_\nu([\![M_2]\!]^{\mathbb{M}}_\nu)$
4. $[\![\lambda x^\alpha.M]\!]^{\mathbb{M}}_\nu$ is the function which maps $m \in \mathcal{M}^\alpha$ to $[\![M]\!]^{\mathbb{M}}_{\nu[x^\alpha\leftarrow m]}$.

It is well-known that the semantics of λ-terms is invariant modulo $\beta\eta$-reduction.

3 Recognizable sets of λ-terms

We wish to extend the notion of recognizability that already exists for strings and trees to λ-terms. An abstract way of defining recognizability for strings and trees is to use Myhill-Nerode theorem [10], [11], that describes it in terms of congruence of finite index over strings or trees which is equivalent to describing it in terms of finite algebra for trees or finite semigroups for strings. This approach has been successfully extended in the seminal paper [12] to any abstract algebra. We shall follow this line of work in order to define recognizability for the simply typed λ-calculus. Since the finite full models form the functional closure of finite algebra, we use them so as to extend recognizability to λ-terms.

Definition 1. *Given a HOS Σ and $\alpha \in \mathcal{T}_\Sigma$ a set $\mathcal{R}$ included in Λ^α_Σ is said to be recognizable iff there is a finite and full model $\mathbb{M} = ((\mathcal{M}^\alpha)_{\alpha\in\mathcal{T}_\Sigma}, \rho)$ a finite variable assignment ν and a subset $\mathcal{P}$ of $\mathcal{M}^\alpha$ such that: $\mathcal{R} = \{M | [\![M]\!]^{\mathbb{M}}_\nu \in \mathcal{P}\}$.*

Note that in this definition when ν is chosen to be the empty assignment function then the set $\mathcal{R}$ only contains closed terms. In particular, when Σ is a tree (*resp.* string) signature, and that α is the atomic type o (*resp.* the type $o \to o$) then the set of closed λ-terms that are recognizable correspond exactly to set of recognizable trees (*resp.* strings).

We give some examples of this fact by showing how to represent some recognizable sets of strings or of string as recognizable sets of λ-terms.

Example 1. Let Σ be the tree signature declaring the operators $+$, 1 which are respectively binary and nullary. Let $\mathcal{A}$ be an automaton with the states $\{even; odd\}$, where $even$ is the final state, and the rules:

$$\begin{aligned} + \; even \; even &\to even \\ + \; odd \; odd &\to even \\ + \; even \; odd &\to odd \\ + \; odd \; even &\to odd \\ 1 &\to odd \end{aligned}$$

So as to define the set of trees recognized by $\mathcal{A}$ as a recognizable set of λ-terms it suffices to define $\mathbb{M}$ to be the model of Σ such that $\mathcal{M}_o = \{even; odd\}$ and the interpretation of $+$ and 1 in $\mathbb{M}$ are given by:

$$\begin{aligned} [\![+]\!]^{\mathbb{M}}(even)(even) &= even \\ [\![+]\!]^{\mathbb{M}}(odd)(odd) &= even \\ [\![+]\!]^{\mathbb{M}}(even)(odd) &= odd \\ [\![+]\!]^{\mathbb{M}}(odd)(even) &= odd \\ [\![1]\!] &= odd \end{aligned}$$

It is then easy to check that the set of trees recognized by $\mathcal{A}$ is the set

$$\{M | [\![M]\!]^{\mathbb{M}} = even\}.$$

This shows that the language recognized by $\mathcal{A}$ is a recognizable set of λ-terms.

Example 2. Let Σ be a string signature declaring the letters a and b. If we define a string automaton with states $\{q_{e,e}; q_{e,o}; q_{o,e}; q_{o,o}\}$ whose initial and final states are respectively $q_{e,e}$ and $q_{o,o}$ (the automaton has a unique final state), and with the following transition function:

$$\begin{array}{ll} \delta(a, q_{e,e}) = q_{o,e} & \delta(a, q_{e,o}) = q_{o,o} \\ \delta(a, q_{o,e}) = q_{e,e} & \delta(a, q_{o,o}) = q_{e,o} \\ \delta(b, q_{e,e}) = q_{e,o} & \delta(b, q_{e,o}) = q_{e,e} \\ \delta(b, q_{o,e}) = q_{o,o} & \delta(b, q_{o,o}) = q_{o,e} \end{array}$$

Then we let $\mathbb{M}$ be the model of Σ generated by the set $\mathcal{M}_o = \{q_{e,e}; q_{e,o}; q_{o,e}; q_{o,o}\}$, we interpret letters as follows:

$$\begin{array}{ll} [\![a]\!]^{\mathbb{M}}(q_{e,e}) = q_{o,e} & [\![a]\!]^{\mathbb{M}}(q_{o,o}) = q_{e,o} \\ [\![a]\!]^{\mathbb{M}}(q_{e,e}) = q_{o,e} & [\![a]\!]^{\mathbb{M}}(q_{e,o}) = q_{o,o} \\ [\![b]\!]^{\mathbb{M}}(q_{e,o}) = q_{e,e} & [\![b]\!]^{\mathbb{M}}(q_{e,e}) = q_{e,o} \\ [\![b]\!]^{\mathbb{M}}(q_{o,o}) = q_{o,e} & [\![b]\!]^{\mathbb{M}}(q_{o,e}) = q_{o,o} \end{array}$$

Then it is easy to check that the set of strings recognized by the automaton is $\{w | [\![/w/]\!]^{\mathbb{M}}(q_{o,o}) = q_{e,e}\}$ this shows that the set $\{/w/ | [\![/w/]\!]^{\mathbb{M}} \in \mathcal{N}\}$ where $\mathcal{N} = \{f \in \mathcal{M}_{o \to o} | f(q_{o,o}) = q_{e,e}\}$ is recognizable.

What the previous examples illustrate is that the notion of recognizability we define on λ-terms is an extension of the notions of recognizability for strings and trees in the sense that a set of ranked trees built on a ranked alphabet Σ, $\mathcal{R}$ is recognizable if and only if it is a recognizable set of λ-terms; similarly a set of string, built on an alphabet Σ, $\mathcal{R}$ is recognizable if and only if the set

$$\{M \in \Lambda_{\Sigma}^{o \to o} | M \text{is closed and} M =_{\beta\eta} /w/ \text{ for some } w \in \mathcal{R}\}$$

is a recognizable set of λ-terms.

The result by Loader [7] shows that in general it is undecidable to check whether a recognizable set is empty. But as soon as the finite and full model and the assignment function are given we can check whether a term is in the set. In what follows we give a mechanical way (corresponding to automata for trees or strings) to define recognizable sets and check whether a certain term belongs to that set.

A classical and simple example of recognizable set of trees being the set of true boolean formulae, we exemplify the notion of recognizability for λ-terms with the set of true Quantified Boolean Formulae (QBF). For this it suffices to use a HOS B whose constants are: $\wedge : b^2 \to b$, $\vee : b^2 \to b$, $\neg : b \to b$, $\mathbf{1} : b$, $\mathbf{0} : b$, $\forall : (b \to b) \to b$ and $\exists : (b \to b) \to b$. We use a finite model $\mathbb{B} = ((\mathcal{B}^{\alpha})_{\alpha \in \mathcal{T}_B}, \rho)$ such that $\mathcal{B}^b = \{0; 1\}$ and ρ associates the obvious functions to the constants of B. Then the set of terms representing a true QBF is the set of closed λ-terms of B^b which are interpreted as 1 in $\mathbb{B}$ and therefore this set is recognizable.

4 Automata characterizing recognizable sets

We here generalize the notion of automata for trees and strings in order to obtain a mechanical definition of recognizability for λ-terms. Our notion of automaton is based on the notion of *Higher Order Intersection Signature (HOIS)* introduced in [3] which, in turn, is based on intersection types [13]. A HOIS is a tuple $\Pi = (\Sigma, I, \iota, \chi)$ where Σ is a HOS, I is a finite set of *atomic intersection types*, ι is a function from I to the atomic types of Σ, χ is a function that associates to every element of $\mathcal{C}$ a subset of $\cap_{\Pi}^{\tau(c)}$ where $(\cap_{\Pi}^{\alpha})_{\alpha \in \mathcal{T}_{\Sigma}}$ is the smallest family verifying:

1. for α and atomic type $\cap_{\Pi}^{\alpha} = \iota^{-1}(\alpha)$,
2. $\cap_{\Pi}^{\alpha \to \beta} = 2^{\cap_{\Pi}^{\alpha}} \times \{\alpha\} \times \cap_{\Pi}^{\beta}$ (we write 2^P, the powerset of the set P)

A trivial induction on the structure of α shows that for each type α, the set $\cap_{\Pi}^{\alpha}$ is finite.

We now define the type system that allows to associate types to λ-terms. Given a HOIS $\Pi = (\Sigma, I, \iota, \chi)$, a *$\Pi$-typing environment* (or simply, *typing environment*, when there is no ambiguity) , is a partial mapping from typed variables to $2^{\cap_{\Pi}}$ whose domain is finite and such that $\Gamma(x^{\alpha})$, when it is defined, is included in $\cap_{\Pi}^{\alpha}$. We write $\overline{\Gamma}$ to denote the domain of Γ. *Judgements over Π* are objects of the form $\Gamma \vdash_{\Pi} M : p$ where Γ is a typing environment, M is an element of

Λ_Σ and p is an element of $\cap_\Pi$. Judgements are derived by using the following inference system:

$$\frac{p \in \Gamma(x^\alpha)}{\Gamma \vdash_\Pi x^\alpha : p}\ \textsc{Axiom} \qquad \frac{p \in \chi(c)}{\Gamma \vdash_\Pi c : p}\ \textsc{Const} \qquad \frac{\Gamma \vdash_\Pi M : p \quad p \sqsubseteq^\alpha_\Pi q}{\Gamma \vdash_\Pi M : q}\ \textsc{Sub}$$

$$\frac{\Gamma, x^\alpha : S \vdash_\Pi M : p}{\Gamma \vdash_\Pi \lambda x^\alpha.M : (S, \alpha, p)}\ \textsc{Abst}$$

$$\frac{\Gamma \vdash_\Pi M : (S, \alpha, p) \quad N \in \Lambda^\alpha_{\Sigma,\overline{\Gamma}} \quad \forall q \in S.\ \Gamma \vdash_\Pi N : q}{\Gamma \vdash (MN) : p}\ \textsc{App}$$

Where the relation $\sqsubseteq^\alpha_\Pi$ is defined as follows:

$$\frac{i \in \iota(\alpha)}{i \sqsubseteq^\iota_\Pi i} \qquad \frac{T \subseteq \cap^\alpha_\Pi \quad \forall p \in S. \exists q \in T. q \sqsubseteq^\alpha_\Pi p}{T \trianglelefteq^\alpha_\Pi S} \qquad \frac{S \trianglelefteq^\alpha_\Pi T \quad q \sqsubseteq^\beta_\Pi p}{(T, \alpha, q) \sqsubseteq^{\alpha\to\beta}_\Pi (S, \alpha, p)}$$

Notice that the rule App, has two premises, concerning N. The reason of being of the premise $N \in \Lambda^\alpha_{\Sigma,\overline{\Gamma}}$ is that when M has an intersection type of the form $(\emptyset, \alpha, p)$, the premise $\forall q \in S.\ \Gamma \vdash_\Pi N : q$ is trivially true and without such a premise we would derive judgments on terms which would not be simply typed.

We will use the notation $\Gamma \vdash_\Pi M : S$ where S is a subset of $\cap^\alpha_\Pi$ to denote the all the judgements of the form $\Gamma \vdash_\Pi M : p$ where p in S. In the same spirit, given S and T that are respectively subsets of $\cap^\alpha_\Pi$ and of $\cap^\beta_\Pi$, we write (S, α, T) to denote the subset of $\cap^{\alpha\to\beta}_\Pi$, $\{(S, \alpha, p) | p \in T\}$.

We now give the principal properties of that system. The proofs of those properties can be found in [3].

Theorem 1 (subject reduction). *If $\Gamma \vdash_\Pi M : p$ is derivable and $M \xrightarrow{*}_{\beta\eta} N$ then $\Gamma \vdash_\Pi N : p$ is derivable.*

Notice that this Theorem would only hold for β-reduction without the use of the rule Sub.

Theorem 2 (subject expansion). *If $M \in \Lambda^\alpha_\Sigma$, $M \xrightarrow{*}_{\beta\eta} N$ and $\Gamma \vdash_\Pi N : p$ is derivable then $\Gamma \vdash_\Pi M : p$ is derivable.*

Theorem 3 (Singleton). *Given $M \in \Lambda^\alpha_\Sigma$, there is Π, Γ and $S \subseteq \cap^\alpha_\Pi$ such that given $N \in \Lambda^\alpha_\Sigma$, $\Gamma \vdash_\Pi N : S$ is derivable if and only if $M =_{\beta\eta} N$.*

This Singleton Theorem, requires some comments. We proved it referring to coherence theorems such as the one proved in [14]. It is also related to a Theorem proved by Statman [8], [9], since we will see that HOIS and finite full models can be represented one in the other.

Since, with intersection type we can represent graphs of functions, the set of λ-terms that are interpreted as a certain element in a finite full model are exactly the λ-terms that verify a certain judgement.

Theorem 4. *Given a HOS Σ, a finite model of Σ, $\mathbb{M} = (\mathcal{M}^\alpha, \rho)$, a finite variable assignment ν over $\mathbb{M}$ and an element e of $\mathcal{M}^\alpha$ then there is a HOIS over Σ, Π, a typing environment Γ and a subset S of $\cap^\alpha_\Pi$ such that for every λ-term M the two following properties are equivalent:*

1. $[\![M]\!]^{\mathbb{M}}_\nu = e$
2. $\Gamma \vdash_\Pi M : S$

A consequence of the previous theorem is that we can obtain the undecidability result by [6] about the emptiness of intersection type as a corollary of the undecidability of λ-decidability [7]. The proof of this Theorem and its consequence are investigated in greater details in the second part of this document.

We now prove the converse of the previous theorem, that is, typability properties in HOIS can be seen as properties of interpretations in finite full models. The principle of this proof is to interpret intersection types as functions operating over intersection types.

We define the operator **app** which maps, for all α and β, $2^{\cap^{\alpha\to\beta}_\Pi} \times 2^{\cap^\alpha_\Pi}$ to $2^{\cap^\beta_\Pi}$. It is defined by: $\mathbf{app}(S,T) = \{p | \exists (Q, \alpha, p) \in S. T \trianglelefteq Q\}$

The finite model in which we will interpret intersection types built over Π is $\mathbb{M}_\Pi = ((\mathcal{M}^\alpha)_{\alpha\in\mathcal{I}_\mathcal{A}}, \rho)$ where for α atomic we let $\mathcal{M}^\alpha$ be $2^{\iota^{-1}(\alpha)}$. The definition of ρ requires an auxiliary function $\mathcal{F}^\alpha$ that sends subsets of $\cap^\alpha_\Pi$ to subsets of $\mathcal{M}^\alpha$ and that verifies:

1. for α atomic and S included in $\cap^\alpha_\Pi$ we let $\mathcal{F}^\alpha(S) = \{T \subseteq \cap^\alpha_\Pi | S \subseteq T\}$,
2. for S included in $\cap^{\alpha\to\beta}_\Pi$ we let

$$\mathcal{F}^{\alpha\to\beta}(S) = \{h \in \mathcal{M}^{\alpha\to\beta} | \forall T \subseteq \cap^\alpha_\Pi . \forall g \in \mathcal{F}^\alpha(T). h(g) \in \mathcal{F}^\beta(\mathbf{app}(S,T))\}$$

It is easy to verify that for every S included in $\cap^\alpha_\Pi$, the set $\mathcal{F}^\alpha(S)$ is not empty. We choose $\rho(c)$ as an element of $\mathcal{F}^{\tau(c)}(\chi(c))$. We then have the following theorem.

Theorem 5. *Given a HOS Σ, a HOIS Π over Σ, Γ and S a subset of $\cap^\alpha_\Pi$, we set $\nu(x^\alpha)$ to be an element of $\mathcal{F}_\alpha(\Gamma(x^\alpha))$, then the two following properties are equivalent:*

1. $\Gamma \vdash_\Pi M : S$
2. $[\![M]\!]^{\mathbb{M}_\Pi}_\nu$ *belongs to* $\mathcal{F}_\alpha(S)$

The Theorems 4 and 5 relate finite models and typability in HOIS. This leads us to the definition of a generalized notion of automaton, *typing-automata*.

Definition 2. *A* typing-automaton*, $\mathcal{A}$, over a HOS Σ is a tuple $(\alpha, \Pi, \Gamma, \{S_1; \dots S_n\})$ where: $\alpha \in \mathcal{T}_\Sigma$, Π is a HOIS over Σ, Γ is a Π-typing environment, for all i in $\{1; \dots; n\}$, S_i is a subset of $\cap^\alpha_\Pi$. The language defined by $\mathcal{A}$ is*

$$L(\mathcal{A}) = \{M | \exists i \in \mathbb{N}.\ \Gamma \vdash_\Pi M : S_i\}$$

Using Theorems 4 and 5 we get:

Theorem 6. *A language of λ-terms L is recognizable if and only if there is a typing-automaton $\mathcal{A}$ such that $L = L(\mathcal{A})$.*

5 Closure properties

5.1 Boolean closure

In this section we shall quickly outline how to construct of typing-automata for the boolean closure properties of recognizable sets of λ-terms. Interestingly these constructions can be seen as generalizations of the usual constructions that are used for tree/string-automata. For example, concerning the intersection of two recognizable languages, we can construct the product of two typing-automata. We first start by defining the product of two HOIS.

Definition 3. *Given $\Pi_1 = (\Sigma, I_1, \iota_1, \chi_1)$ and $\Pi_2 = (\Sigma, I_2, \iota_2, \chi_2)$ we define the HOIS $\Pi_1 \otimes \Pi_2$ to be (Σ, I, ι, χ) where:*

1. *I is a subset of $I_1 \times I_2$ which is equal to $\{(p_1, p_2) | \iota_1(p_1) = \iota_2(p_2)\}$,*
2. *$\iota((p_1, p_2)) = \iota_1(p_1)$, note that by definition of I, $\iota((p_1, p_2)) = \iota_2(p_2)$,*
3. *$\chi(c) = \{p_1 \otimes p_2 | p_1 \in \chi_1(c)$ and $p_2 \in \chi_2(c)\}$.*

where given p_1 in $\cap^{\alpha}_{\Pi_1}$ and p_2 in $\cap^{\alpha}_{\Pi_2}$ we define $p_1 \otimes p_2$ by:

1. *if α is atomic then $p_1 \otimes p_2 = (p_1, p_2)$*
2. *if $\alpha = \alpha_1 \to \alpha_2$ then $p_1 = (S_1, \alpha_1, q_1)$ and $p_2 = (S_2, \alpha_2, q_2)$ and $p_1 \otimes p_2 = (S_1 \otimes S_2, \alpha_1, q_1 \otimes q_2)$ where $S_1 \otimes S_2 = \{r_1 \otimes r_2 | r_1 \in S_1$ and $r_2 \in S_2\}$*

If we define the product of two typing environment Γ and Δ to be $\Gamma \otimes \Delta$ such that $\Gamma \otimes \Delta(x) = \Gamma(x) \otimes \Delta(x)$, we can prove the following property:

Theorem 7. *The judgements $\Gamma \vdash_{\Pi_1} M : P$ and $\Delta \vdash_{\Pi_2} M : Q$ are derivable if and only if the judgement $\Gamma \otimes \Delta \vdash_{\Pi_1 \otimes \Pi_2} M : P \otimes Q$ is derivable.*

This allows us to define the product $\mathcal{A} \otimes \mathcal{B}$ of two typing-automata $\mathcal{A}$ and $\mathcal{B}$.

Definition 4. *Given two typing-automata over some HOS Σ, $\mathcal{A} = (\alpha, \Pi_1, \Gamma, T_1)$ and $\mathcal{B} = (\alpha, \Pi_2, \Delta, T_2)$, we let $\mathcal{A} \otimes \mathcal{B}$ be $(\alpha, \Pi_1 \otimes \Pi_2, \Gamma \otimes \Delta, T_1 \otimes T_2)$ where $T_1 \otimes T_2$ is the set $\{S_1 \otimes S_2 | S_1 \in T_1$ and $S_2 \in T_2\}$.*

Theorem 8. *Given two typing automata of the same type over some HOS Σ, $\mathcal{A}$ and $\mathcal{B}$ we have $L(\mathcal{A} \otimes \mathcal{B}) = L(\mathcal{A}) \cap L(\mathcal{B})$.*

The closure under complement of the class of recognizable sets of λ-terms, is a direct consequence of its definition in terms of finite models. Interestingly, if one wishes to construct the typing-automaton recognizing the complementary language of a given typing-automaton, then one would use the construction that serves in Theorem 5 which on a tree or string automaton would corresponds to *determinization*. This induces a notion of deterministic typing-automata that grasps the notion of recognizability, and corresponding to the fact that intersection types correspond to partial function over the finite model generated by the atomic intersection types.

5.2 Homomorphisms

It is well-known that recognizable sets of strings are closed under arbitrary homomorphisms while recognizable sets of trees are closed under linear homomorphisms. We will see that recognizable sets of λ-terms are not even closed under relabeling. This has the consequence, that Monadic Second Order Logic (MSO) over the structure of normal λ-terms is not grasped by our notion of recognizability, since relabelings allow to represent set quantification. On the other hand, alike strings and trees, recognizable sets of λ-terms are closed under arbitrary inverse homomorphisms.

We now turn to show that recognizable sets of λ-terms are not closed under relabeling. In order to show this we use the following signature $\Sigma = \{\forall : (b \to b) \to b, \wedge : b \to b \to b, \vee : b \to b \to b, \neg : b \to b, \lhd : b \to b \to b, \rhd : b \to b \to b\}$. Since terms built on Σ are usual boolean expressions, we shall use the standard notation for those expressions instead of the λ-term notation. Thus we shall write $\forall x.t$, $t_1 \wedge t_2$ and $t_1 \vee t_2$ instead of $\forall(\lambda x.t)$, $\wedge t_1\, t_2$ and $\vee t_1\, t_2$. The terms built on Σ are interpreted in a finite model $\mathbb{B} = ((\mathcal{B}^\alpha)_{\alpha\in\mathcal{T}_\Sigma}, \rho)$ where $\mathcal{B}^b = \{0; 1\}$ and ρ interprets the usual boolean connectives and quantifiers ($\wedge$, $\vee$, $\neg$ and $\forall$) with their usual truth tables and ρ interprets the connectives $\lhd$ and $\rhd$ as the functions such that $\rho(\lhd)xy = x$ and $\rho(\rhd)xy = y$. By definition the set $\mathcal{T}$ of closed terms whose semantic interpretation in $\mathbb{B}$ is 1 is recognizable.

We use a relabeling $\mathcal{H}$ which maps the terms built on Σ to terms built on $\Sigma' = \{\forall : (b \to b) \to b, \wedge : b \to b \to b, \vee : b \to b \to b, \neg : b \to b, \bowtie\colon b \to b \to b\}$ so the constants $\forall$, $\wedge$, $\vee$ and $\neg$ are mapped to themselves by $\mathcal{H}$ and $\lhd$ and $\rhd$ are both mapped to $\bowtie$.

We let $\Leftrightarrow$ be the λ-term $\lambda xy.(x \wedge y) \vee (\neg x \wedge \neg y)$; as for the other connective, we adopt an infix notation, *i.e.* we shall write $t_1 \Leftrightarrow t_2$ instead of $\Leftrightarrow t_1\, t_2$.

As the connective $\lhd$ (*resp.* $\rhd$) takes the value of its left (*resp.* right) argument, if f and g are terms whose free variables are $x_1, \ldots, x_n$, then we have the following identities $[\![\forall x_1 \ldots \forall x_n.(\lhd f\, g) \Leftrightarrow f]\!]^{\mathbb{B}} = 1$ and $[\![\forall x_1 \ldots \forall x_n.(\rhd f\, g) \Leftrightarrow g]\!]^{\mathbb{B}} = 1$.

The closed term $\lambda x_1^b \ldots x_n^b.t$ built on Σ can be interpreted as a function from $\{0; 1\}^n$ to $\{0; 1\}$ (modulo curryfication) in $\mathbb{B}$, *i.e.* an n-ary boolean function. For a given n there are 2^{n+1} such functions and we know that for each such function f we can build, using only $\wedge$, $\vee$ and $\neg$, a term $\widetilde{f}$ such that $[\![\lambda x_1 \ldots x_n.\widetilde{f}]\!]^{\mathbb{B}} = f$. Given $\mathcal{F} = \{f_1; \ldots; f_p\}$ a set of such functions, we write $[\mathcal{F}]$ the term $\bowtie \widetilde{f_1}(\bowtie \widetilde{f_2}(\ldots(\bowtie \widetilde{f_{p-1}}\widetilde{f_p})\ldots))$. Remark that for all i in $\{1; \ldots; p\}$, there is H_i such that $\mathcal{H}(H_i) = [\mathcal{F}]$ and $[\![\forall(x_1 \ldots \forall(x_n.H_i \Leftrightarrow \widetilde{f_i})\ldots)]\!]^{\mathbb{B}} = 1$ and thus $\forall(x_1 \ldots \forall(x_n.[\mathcal{F}] \Leftrightarrow \widetilde{f_i})$ is in $\mathcal{H}(\mathcal{T})$. Furthermore for every H such that $\mathcal{H}(H) = \mathcal{F}$ there is i in $\{1; \ldots; p\}$ such that $[\![\forall(x_1 \ldots \forall(x_n.H \Leftrightarrow \widetilde{f_i})\ldots)]\!]^{\mathbb{B}} = 1$. If we suppose that $\mathcal{H}(\mathcal{T})$ is recognizable, then there is a finite model $\mathbb{M} = ((\mathcal{M}^\alpha)_{\alpha\in\mathcal{T}_\Sigma}, \rho)$ and a subset $\mathcal{N}$ of $\mathcal{M}^b$ such that the closed terms M are in $\mathcal{H}(\mathcal{T})$ if and only if $[\![M]\!]^{\mathbb{M}} \in \mathcal{N}$; we assume that $\mathcal{M}^b$ contains q elements. Each closed term $\lambda x_1^b \ldots x_n^b.M$ built on Σ' is interpreted in $\mathbb{M}$ as a function from $\{1; \ldots; q\}^n$ to $\{1; \ldots; q\}$ (modulo curryfication). We are going to show that for every sets

of n-ary boolean functions $\mathcal{F}$ and $\mathcal{G}$, it is necessary that $[\![\lambda x_1^b \dots x_n^b.[\mathcal{F}]]\!]^{\mathbb{M}}$ and $[\![\lambda x_1^b \dots x_n^b.[\mathcal{G}]]\!]^{\mathbb{M}}$ are different when $\mathcal{F}$ and $\mathcal{G}$ are different. Indeed, if $\mathcal{F}$ and $\mathcal{G}$ are different, we can assume without loss of generality that $\mathcal{F}$ is not empty, and then there is a boolean function f which is in $\mathcal{F}$ and which is not in $\mathcal{G}$. Since there is H such that $\mathcal{H}(H) = [\mathcal{F}]$ and $[\![\forall x_1. \dots \forall x_n.H \Leftrightarrow \widetilde{f}]\!]^{\mathbb{B}} = 1$, then $\forall x_1. \dots \forall x_n.[\mathcal{F}] \Leftrightarrow \widetilde{f}$ is in $\mathcal{H}(\mathcal{T})$. But for an n-ary boolean g, there is an H' such that $\mathcal{H}(H') = [\mathcal{G}]$ and $[\![\forall x_1. \dots \forall x_n.H' \Leftrightarrow \widetilde{g}]\!]^{\mathbb{B}} = 1$ iff g is in $\mathcal{G}$. Thus the term $\forall x_1. \dots \forall x_n.[G] \Leftrightarrow \widetilde{f}$ is not in $\mathcal{H}(\mathcal{T})$ and $[\![\lambda x_1^b \dots x_n^b.[\mathcal{F}]]\!]^{\mathbb{M}}$ is different from $[\![\lambda x_1^b \dots x_n^b.[\mathcal{G}]]\!]^{\mathbb{M}}$. But there are $2^{2^{n+1}}$ sets of n-ary boolean functions while there are q^{n+1} functions from $\{1; \dots; q\}^n$ to $\{1; \dots; q\}$ and thus for n sufficiently big, it is not possible to verify that $[\![\lambda x_1^b \dots x_n^b.[\mathcal{F}]]\!]^{\mathbb{M}}$ and $[\![\lambda x_1^b \dots x_n^b.[\mathcal{G}]]\!]^{\mathbb{M}}$ are different when $\mathcal{F}$ and $\mathcal{G}$ are different. Therefore, $\mathcal{H}(\mathcal{T})$ is not a recognizable set. This implies that the class of recognizable sets of λ-terms is not closed under relabeling.

While there seems to be no interesting class of homomorphisms under which our notion of recognizability is closed, we can show that recognizable sets of λ-terms are closed under inverse homomorphism.

Theorem 9. *Given Σ_1, Σ_2 two HOS and $\mathcal{H}$ a homomorphism between Σ_1 and Σ_2, if R is a recognizable set of Σ_2 then $\mathcal{H}^{-1}(R) \cap \Lambda^{\alpha}_{\Sigma,V}$ is also recognizable.*

Recognizable sets contain only λ-terms of a given type and there is no reason why $\mathcal{H}^{-1}(R)$ is a set containing terms having all the same type. So intersecting $\mathcal{H}^{-1}(R)$ with set set of the form $\Lambda^{\alpha}_{\Sigma,V}$ is necessary.

6 Some applications of recognizability

We here quickly review some direct applications of the notion of recognizability in the simply typed λ-calculus.

6.1 Parsing

Theorem 9 gives a very simple definition of parsing for many formalisms. Indeed in formalisms, such as Context Free Grammars, Tree Adjoining Grammars, Multiple Context Free Grammars, Parallel Multiple Context Free Grammars *etc...* can be seen as the interpretation of trees via homomorphism (see [15]). Thus these grammars can be seen a 4-tuple $(\Sigma_1, \Sigma_2, \mathcal{H}, S)$ where Σ_1 is a multi-sorted tree signature, Σ_2 is a string signature, $\mathcal{H}$ is a homomorphism from Σ_1 to Σ_2 and S is the type of the trees that are considered as analyses. Thus if we want to parse a word w we try to find the set $\{M \in \Lambda^S_{\Sigma_1} | M \text{ is closed and } \mathcal{H}(M) =_{\beta\eta} w\}$ which is actually $\mathcal{H}^{-1}(\{w\})$. But we know from Theorem 3 that $\{w\}$ is a recognizable set and thus parsing amounts to compute the inverse homomorphic image of a recognizable set. This gives a new proof of the theorem of [16] which proves that the set of parse trees of a sentence in a context free grammars is a

recognizable set, and it furthermore generalizes the result to a wide family of formalisms. Moreover, this view on parsing also applies to grammars generating tree or λ-terms, it also shows that parsing a structure is similar to parsing recognizable sets. Parsing recognizable sets instead of singleton structures has the advantage that it allows to parse ambiguous inputs, such as noisy phonetic transcriptions, or ambiguous tagging of sentences...

6.2 Higher order matching

The γ-higher-order matching problem (γ-HOM), with $\gamma \in \{\beta; \beta\eta\}$, consists in solving an equation of the form $M \stackrel{?}{=}_\gamma N$ where N is a closed term. A solution of such an equation is a substitution σ such that $M.\sigma =_\gamma N$. Using the extraction Lemma of [3], and Theorem 3, it is easy to see that the solutions of $\beta\eta$-HOM form finite unions of cartesian products of recognizable sets. Observing this, allows us to obtain in an alternative way the relation between λ-definability and $\beta\eta$-HOM showed in [8]. Furthermore, we can easily obtain the result that $\beta\eta$-HOM is decidable (see [17]) when the terms in a solution are *arity bounded*, *i.e.* under the constraint that the number of variables that can be free in a subterm is bounded by some number k. Indeed, because of the subformula property and the bound on the number of free variables, arity-bounded terms of a given type can all be represented with finitely many combinators; this means that we can represent those terms in a tree-HOS Σ and recover them with a homomorphism $\mathcal{H}$. Thus, the set S of terms that are solution of arity bounded $\beta\eta$-HOM can be effectively represented as a recognizable set of trees, namely $\mathcal{H}^{-1}(S)$, the emptiness of recognizable sets of trees being decidable this gives the decidability of arity bounded $\beta\eta$-HOM. In particular, this leads to the decidability of arity bounded $\beta\eta$-HOM. Since arity-bounded $\beta\eta$-HOM is more general than 3^{rd} and 4^{th} order $\beta\eta$-HOM [17], this technique sheds some light on the results obtained by [18] that relate the solutions of these special cases to tree automata. Contrary to most approach to HOM, the one we use is completely direct, we do not need to transform the problem within a set of interpolation equations.

β-HOM [19] is undecidable while $\beta\eta$-HOM seems to be decidable [20]. But there is no satisfying explanation on the difference between β-HOM and $\beta\eta$-HOM so as to account satisfactorily of that difference. But as we have seen, intersection types make a discrimination between β-reduction and $\beta\eta$-reduction with the rule Sub, without which the subject reduction Theorem does not hold for $\beta\eta$-reduction. Thus intersection types seem to be a good tool to investigate this problem.

7 Conclusion and future work

We have defined a notion of recognizability for the λ-calculus that naturally extends recognizability for trees or strings. We have exhibited the closure properties of this notion and showed how it could be exploited to understand parsing of the

higher order matching problem. Contrary to strings and trees where recognizability comes with three kinds of characterization, a mechanical one (automata), an algebraic one and a logical one (Monadic Second Order Logic, MSOL), here our notion only comes with a mechanical and an algebraic characterization. It seems difficult to come up with a logical characterization since this notion is not closed under relabeling. And closure under relabeling is central to represent quantification in MSOL. As we wish to use this notion of recognizability so as to describe particular sets of λ-terms, it would be nice to obtain a connection with some logic. First-order logic would be a first step. A more general question would be whether there is a logic that exactly corresponds to this notion of recognizability.

Another question is to characterize the restrictions under which the emptiness of recognizable sets is decidable. Theorem 9 gives a positive answer when the terms are bound to be generated with a finite set of combinators since it reduces this emptiness problem to the emptiness problem of some recognizable set of trees. But we do not know whether there are weaker constraints for which this holds. When we look at the situation for graphs, there is no class of graphs [21] which can be generated only with infinitely many combinators (this means that the class of graphs has an infinite treewidth) for which this emptiness problem is decidable. Thus, this question can be related to the definition of a suitable notion for normal λ-terms that would be similar to treewidth for graphs.

Finally we hope that the notion of recognizability for λ-terms can be of interest in the study of trees generated by higher-order programming schemes. It has been showed that those trees had a decidable MSO theory[22]. It is likely that intersection types should be more adapted to conduct this proof, and yield to new techniques.

References

1. de Groote, P.: Towards abstract categorial grammars. In for Computational Linguistic, A., ed.: Proceedings 39th Annual Meeting and 10th Conference of the European Chapter, Morgan Kaufmann Publishers (2001) 148–155
2. Montague, R.: Formal Philosophy: Selected Papers of Richard Montague. Yale University Press, New Haven, CT (1974)
3. Salvati, S.: On the membership problem for Non-linear Abstract Categorial Grammars. In Muskens, R., ed.: Proceedings of the Workshop on New Directions in Type-theoretic Grammars (NDTTG 2007), Dublin, Ireland, Foundation of Logic, Language and Information (FoLLI) (August 2007) 43–50
4. Lambek, J.: The mathematics of sentence structure. American Mathematical Monthly **65** (1958) 154–170
5. Moortgat, M.: Categorial Investigations: Logical & Linguistic Aspects of the Lambek Calculus. Foris Pubns USA (1988)
6. Urzyczyn, P.: The emptiness problem for intersection types. J. Symb. Log. **64**(3) (1999) 1195–1215
7. Loader, R.: The undecidability of λ-definability. In Anderson, C.A., Zeleny, M., eds.: Logic, Meaning and Computation: Essays in memory of Alonzo Church. Kluwer (2001) 331–342

8. Statman, R.: Completeness, invariance and λ-definability. Journal of Symbolic Logic **47**(1) (1982) 17–26
9. Statman, R., Dowek, G.: On statman's finite completeness theorem. Technical Report CMU-CS-92-152, University of Carnegie Mellon (1992)
10. Myhill, J.: Finite automata and the representation of events. Technical Report WADC TR-57-624, Wright Patterson Air Force Base, Ohio, USA (1957)
11. Nerode, A.: Linear automaton transformations. In: Proceedings of the American Mathematical Society. Volume 9., American Mathematical Society (1958) 541–544
12. Mezei, J., Wright, J.: Algebraic automata and context-free sets. Informcation and Control **11** (1967) 3–29
13. Dezani-Ciancaglini, M., Giovannetti, E., de' Liguoro, U.: Intersection Types, Lambda-models and Böhm Trees. In: MSJ-Memoir Vol. 2 "Theories of Types and Proofs". Volume 2. Mathematical Society of Japan (1998) 45–97
14. Babaev, A.A., Soloviev, S.V.: Coherence theorem for canonical maps in cartesian closed categories. Journal of Soviet Mathematics **20** (1982)
15. de Groote, P., Pogodalla, S.: On the expressive power of abstract categorial grammars: Representing context-free formalisms. Journal of Logic, Language and Information **13**(4) (2005) 421–438
16. Thatcher, J.W.: Characterizing derivation trees of context-free grammars through a generalization of finite automata theory. Journal of Computer and System Sciences **1**(4) (December 1967) 317–322
17. Schmidt-Schauß, M.: Decidability of arity-bounded higher-order matching. In: CADE-19. LNCS 2741, Springer (2003) 488–502
18. Comon, H., Jurski, Y.: Higher-order matching and tree automata. In: CSL. (1997) 157–176
19. Loader, R.: Higher order β matching is undecidable. Logic Journal of the IGPL **11**(1) (2003) 51–68
20. Stirling, C.: A game-theoretic approach to deciding higher-order matching. In Bugliesi, M., Preneel, B., Sassone, V., Wegener, I., eds.: ICALP (2). Volume 4052 of Lecture Notes in Computer Science., Springer (2006) 348–359
21. Robertson, N., Seymour, P.D.: Graph minors. v. excluding a planar graph. J. Comb. Theory, Ser. B **41**(1) (1986) 92–114
22. Ong, C.H.L.: On model-checking trees generated by higher-order recursion schemes. In: LICS, IEEE Computer Society (2006) 81–90

Part II

Correspondence between finite models and intersection types

Introduction

Consider the simply typed λ-calculus on simple types $\mathbb{T}^0$ with one ground type 0. Recall that a hereditarily finite full model of simply typed λ-calculus is a collection of sets $\mathcal{F} = (\mathcal{F}_A)_{A \in \mathbb{T}^0}$ such that $\mathcal{F}_0 \neq \emptyset$ is finite and $\mathcal{F}_{A \to B} = \mathcal{F}_B^{\mathcal{F}_A}$ (i.e. the set of functions from $\mathcal{F}_A$ to $\mathcal{F}_B$) for all simple types A, B. An element $f \in \mathcal{F}_A$ is λ-*definable* whenever, for some closed λ-term M having type A, we have $[\![M]\!] = f$, where $[\![M]\!]$ denotes the interpretation of M in $\mathcal{F}$. The following question, raised by Plotkin in [7], is known as the Definability Problem:

DP: "Given an element f of any hereditarily finite full model, is f λ-definable?"

A natural restriction considered in the literature [5,6] is the following:

DP_n: "Given an element f of $\mathcal{F}_n$, is f λ-definable?"

where $\mathcal{F}_n$ (for $n \geq 1$) denotes the unique (up to isomorphism) full model whose ground set $\mathcal{F}_0$ has n elements. Statman's conjecture stating that DP is decidable [9] was refuted by Loader [6], who proved in 1993 (but published in 2001) that DP_n is undecidable for every $n > 6$. Such a result was then strengthened by Joly, who showed in [5] that DP_n is undecidable for all $n > 1$.

Theorem 10. *1.* **(Loader)** *The Definability Problem is undecidable.*
2. **(Loader/Joly)** *DP_n is undecidable for every $n > 6$ (resp. $n > 1$).*

Consider now the λ-calculus endowed with the intersection type system CDV (Coppo-Dezani-Venneri [4]) based on a countable set $\mathbb{A}$ of atomic types. Recall that an intersection type σ is *inhabited* if $\vdash_\wedge M : \sigma$ for some closed λ-term M.

The Inhabitation Problem for this type theory is formulated as follows:

IHP: "Given an intersection type σ, is σ inhabited?"

We will also be interested in the following restriction of IHP:

IHP_n: "Given an intersection type σ with at most n atoms, is σ inhabited?"

In 1999, Urzyczyn [10] proved that IHP is undecidable for suitable intersection types, called "game types" in [3], and thus for the whole CDV. His idea was to prove that solving IHP for a game type σ is equivalent to winning a suitable "tree game" G. An arbitrary number of atoms may be needed since, in the Turing-reduction, the actual amount of atoms in σ is determined by G.

Theorem 11 (Urzyczyn).

1. *The Inhabitation Problem is undecidable.*
2. *The Inhabitation Problem for game types is undecidable.*

The undecidability of DP and that of IHP are major results presented thoroughly in [3]. In the proof these problems are reduced to well-known undecidable problems (and eventually to the Halting problem). However, the instruments used to achieve these results are very different — the proof by Loader proceeds by reducing DP to the two-letter word rewriting problem, while the proof by Urzyczyn reduces IHP to the emptiness problem for queue automata (through a series of reductions). The fact that these proofs are different is not surprising since the two problems, at first sight, really *look* unrelated.

Our main contribution is to show that DP and IHP are actually Turing-equivalent, by providing a perhaps unexpected link between the two problems. The key ideas behind our constructions are the following. Every intersection $\alpha_1 \wedge \cdots \wedge \alpha_k$ of atoms can be viewed as a set $\{\alpha_1, \ldots, \alpha_k\}$, and every arrow type $\sigma \to \tau$ as a (continuous) step function. Moreover, Urzyczyn's game types follow the structure of simple types. Combining these ingredients we build a continuous model $\mathcal{S} = (\mathcal{S}_A)_{A \in \mathbb{T}^0}$ over a finite set of atomic types, which constitutes a "bridge" between intersection type systems and full models of simply typed λ-calculus. Then, exploiting very natural semantic logical relations, we can study the continuous model, cross the bridge and infer properties of the full model. Our constructions allow us to obtain the following Turing-reductions (recall that if the problem P_1 is undecidable and $P_1 \leq_T P_2$, then also P_2 is undecidable):

(i) Inhabitation Problem for game types $\leq_T$ Definability Problem,
(ii) Definability Problem $\leq_T$ Inhabitation Problem (cf. [8]),
(iii) $\mathrm{DP}_n \leq_T \mathrm{IHP}_n$ (cf. [8]).

Therefore, by (i) and (ii) we get that the undecidability of DP and IHP follows from each other. Moreover, by (iii) and Theorem 10(2) we conclude that IHP_n is undecidable whenever $n > 1$, which is a new result refining Urzyczyn's one.

8 Preliminaries: Some Syntax, Some Semantics

To make this article more self-contained, this section summarizes some definitions and results that we will use later in the paper. Given a set X, we write $\mathcal{P}(X)$ for the set of all subsets of X, and $Y \subseteq_{\mathrm{f}} X$ if Y is a finite subset of X.

8.1 Typed Lambda Calculi

We take untyped λ-calculus for granted together with the notions of closed λ-term, α-conversion, (β-)normal form and strong normalization. We denote by Var the set of variables and by Λ the set of λ-terms. Hereafter, we consider λ-terms up to α-conversion and we adopt Barendregt's variable convention.

We mainly focus on two particular typed λ-calculi (see [3] for more details).

The simply typed λ-calculus *à la Curry* over a single atomic type 0. The set $\mathbb{T}^0$ of *simple types* $A, B, C, \ldots$ is defined in Figure 1(a). *Simple contexts* Δ are partial functions from Var to $\mathbb{T}^0$; we write $\Delta = x_1 : A_1, \ldots, x_n : A_n$ for the

$$\begin{array}{ll} \Lambda: & M,N,P ::= x \mid MN \mid \lambda x.M, \text{ where } x \in \mathrm{Var} \\ \mathbb{T}^0: & A,B,C \ ::= 0 \mid A \to B \\ \mathbb{T}^{\mathbb{A}}_{\wedge}: & \gamma,\rho,\sigma,\tau ::= \alpha \mid \sigma \to \tau \mid \sigma \wedge \tau, \text{ where } \alpha \in \mathbb{A} \end{array}$$

(a) Sets Λ of λ-terms, $\mathbb{T}^0$ of simple types, $\mathbb{T}^{\mathbb{A}}_{\wedge}$ of intersection types over $\mathbb{A}$.

$$\sigma \leq \sigma \ (\text{refl}) \qquad \sigma \wedge \tau \leq \sigma \ (\text{incl}_L) \qquad \sigma \wedge \tau \leq \tau \ (\text{incl}_R)$$

$$(\sigma \to \tau) \wedge (\sigma \to \tau') \leq \sigma \to (\tau \wedge \tau') \quad (\to_\wedge)$$

$$\frac{\sigma \leq \gamma \quad \gamma \leq \tau}{\sigma \leq \tau} \ (\text{trans}) \qquad \frac{\sigma \leq \tau \quad \sigma \leq \tau'}{\sigma \leq \tau \wedge \tau'} \ (\text{glb}) \qquad \frac{\sigma' \leq \sigma \quad \tau \leq \tau'}{\sigma \to \tau \leq \sigma' \to \tau'} \ (\to)$$

(b) Rules defining the subtyping relation $\leq$ on intersection types $\mathbb{T}^{\mathbb{A}}_{\wedge}$.

$$\frac{}{x_1:\sigma_1,\ldots,x_n:\sigma_n \vdash_\wedge x_i:\sigma_i} \ (\text{ax}) \qquad \frac{\Gamma \vdash_\wedge M:\tau \to \sigma \quad \Gamma \vdash_\wedge N:\tau}{\Gamma \vdash_\wedge MN:\sigma} \ (\to_E)$$

$$\frac{\Gamma, x:\sigma \vdash_\wedge M:\tau}{\Gamma \vdash_\wedge \lambda x.M:\sigma \to \tau} (\to_I) \qquad \frac{\Gamma \vdash_\wedge M:\sigma \quad \Gamma \vdash_\wedge M:\tau}{\Gamma \vdash_\wedge M:\sigma \wedge \tau} \ (\wedge_I) \qquad \frac{\Gamma \vdash_\wedge M:\sigma \quad \sigma \leq \tau}{\Gamma \vdash_\wedge M:\tau} \ (\leq)$$

(c) Rules defining the intersection type system CDV.

Fig. 1: Definition of terms, types, subtyping and derivation rules for CDV. The rules for simply typed λ-calculus are obtained from those in (c) leaving out $(\wedge_I)$ and $(\leq)$.

function of domain $\{x_1,\ldots,x_n\}$ such that $\Delta(x_i) = A_i$ for i in $[1;n]$. We write $\Delta \vdash M : A$ if M *has type* A *in* Δ, and we say that such an M is *simply typable*.

The intersection type system CDV over an infinite set $\mathbb{A}$ of atomic types. This system was first introduced by Coppo, Dezani and Venneri [4] to characterize strongly normalizable λ-terms. The set $\mathbb{T}^{\mathbb{A}}_{\wedge}$ of *intersection types* is given in Figure 1(a) and it is partially ordered by the subtyping relation $\leq$ defined in Figure 1(b). We denote by $\simeq$ the equivalence generated by $\leq$. As usual, we may write $\bigwedge_{i=1}^n \sigma_i \to \tau_i$ for $(\sigma_1 \to \tau_1) \wedge \cdots \wedge (\sigma_n \to \tau_n)$.

Contexts $\Gamma = x_1:\tau_1,\ldots,x_n:\tau_n$ are handled as in the simply typed case. We write $\Gamma \vdash_\wedge M : \sigma$ if the judgment can be proved using the rules of Figure 1(c).

As a matter of notation, given two sets Y, Z of intersection types, we let $Y^\wedge = \{\sigma_1 \wedge \cdots \wedge \sigma_n \mid \sigma_i \in Y \text{ for } i \in [1;n]\}$ and $Y \to Z = \{\tau \to \sigma \mid \tau \in Y, \sigma \in Z\}$.

We now present some well known properties of CDV. For their proofs, we refer to [4], [3, Thm. 14.1.7] and [3, Thm. 14.1.9] respectively.

Theorem 12. *A λ-term M is typable in CDV iff M is strongly normalizable.*

Theorem 13 (β-soundness). *For all $k \geq 1$, if $\bigwedge_{i=1}^k \sigma_i \to \rho_i \leq \gamma_1 \to \gamma_2$ then there is a non-empty subset $K \subseteq [1;k]$ such that $\gamma_1 \leq \bigwedge_{i\in K} \sigma_i$ and $\bigwedge_{i\in K} \rho_i \leq \gamma_2$.*

Theorem 14 (Inversion Lemma). *The following properties hold:*

1. *$\Gamma \vdash_\wedge x : \sigma$ iff $\Gamma(x) \leq \sigma$,*

2. $\Gamma \vdash_\wedge MN : \sigma$ *iff there is* ρ *such that* $\Gamma \vdash_\wedge M : \rho \to \sigma$ *and* $\Gamma \vdash_\wedge N : \rho$,
3. $\Gamma \vdash_\wedge \lambda x.M : \sigma$ *iff there is* $n \geq 1$ *such that* $\sigma = \bigwedge_{i=1}^{n} \sigma_i \to \sigma'_i$ *for some* σ_i, σ'_i,
4. $\Gamma \vdash_\wedge \lambda x.M : \sigma \to \tau$ *iff* $\Gamma, x : \sigma \vdash_\wedge M : \tau$.

8.2 Type Structures Modelling the Simply Typed Lambda Calculus

A *typed applicative structure* $\mathcal{M}$ is a pair $((\mathcal{M}_A)_{A \in \mathbb{T}^0}, \bullet)$ where each $\mathcal{M}_A$ is a structure whose carrier is non-empty, and $\bullet$ is a function that associates to every $d \in \mathcal{M}_{A \to B}$ and every $e \in \mathcal{M}_A$ an element $d \bullet e$ in $\mathcal{M}_B$. From now on, we shall write $d \in \mathcal{M}$ to denote $d \in \mathcal{M}_A$ for some A. We say that $\mathcal{M}$ is: *hereditarily finite* if every $\mathcal{M}_A$ has a finite carrier; *extensional* whenever, for all $A, B \in \mathbb{T}^0$ and $d, d' \in \mathcal{M}_{A \to B}$, we have that $d \bullet e = d' \bullet e$ for every $e \in \mathcal{M}_A$ entails $d = d'$.

A *valuation in* $\mathcal{M}$ is any map $\nu_\mathcal{M}$ from Var to elements of $\mathcal{M}$. A valuation $\nu_\mathcal{M}$ *agrees* with a simple context Δ when $\Delta(x) = A$ implies $\nu_\mathcal{M}(x) \in \mathcal{M}_A$. Given a valuation $\nu_\mathcal{M}$ and an element $d \in \mathcal{M}$, we write $\nu_\mathcal{M}[x := d]$ for the valuation $\nu'_\mathcal{M}$ that coincides with $\nu_\mathcal{M}$, except for x, where $\nu'_\mathcal{M}$ takes the value d. When there is no danger of confusion we may omit the subscript $\mathcal{M}$ and write ν.

A *valuation model* $\mathcal{M}$ is an extensional typed applicative structure such that the clauses below define a total interpretation function $[\![\cdot]\!]^\mathcal{M}_{(\cdot)}$ which maps derivations $\Delta \vdash M : A$ and valuations ν agreeing with Δ to elements of $\mathcal{M}_A$:

- $[\![\Delta \vdash x : A]\!]^\mathcal{M}_\nu = \nu(x)$,
- $[\![\Delta \vdash NP : A]\!]^\mathcal{M}_\nu = [\![\Delta \vdash N : B \to A]\!]^\mathcal{M}_\nu \bullet [\![\Delta \vdash P : B]\!]^\mathcal{M}_\nu$,
- $[\![\Delta \vdash \lambda x.N : A \to B]\!]^\mathcal{M}_\nu \bullet d = [\![\Delta, x : A \vdash N : B]\!]^\mathcal{M}_{\nu[x:=d]}$ for every $d \in \mathcal{M}_A$.

When the derivation (resp. the model) is clear from the context we may simply write $[\![M]\!]^\mathcal{M}_\nu$ (resp. $[\![M]\!]_\nu$). For M closed, we simplify the notation further and write $[\![M]\!]$ since its interpretation is independent from the valuation.

The full model over a set $X \neq \emptyset$, denoted by $\mathrm{Full}(X)$, is the valuation model $((\mathcal{F}_A)_{A \in \mathbb{T}^0}, \bullet)$ where $\bullet$ is functional application, $\mathcal{F}_0 = X$ and $\mathcal{F}_{A \to B} = \mathcal{F}_B^{\mathcal{F}_A}$.

The continuous model over a cpo $(D, \leq)$, written $\mathrm{Cont}(D, \leq)$, is the valuation model $((\mathcal{D}_A, \sqsubseteq_A)_{A \in \mathbb{T}^0}, \bullet)$ such that $\bullet$ is functional application and:

- $\mathcal{D}_0 = D$ and $f \sqsubseteq_0 g$ iff $f \leq g$,
- $\mathcal{D}_{A \to B} = [\mathcal{D}_A \to \mathcal{D}_B]$ consisting of the monotone functions from $\mathcal{D}_A$ to $\mathcal{D}_B$ with the pointwise partially ordering $\sqsubseteq_{A \to B}$.

We will systematically omit the subscript A in $\sqsubseteq_A$ when clear from the context.

Note that both $\mathrm{Full}(X)$ and $\mathrm{Cont}(D, \leq)$ are extensional. Moreover, whenever X (resp. D) is finite $\mathrm{Full}(X)$ (resp. $\mathrm{Cont}(D, \leq)$) is hereditarily finite.

Logical relations have been extensively used in the study of semantic properties of λ-calculus (see [2] for a survey). As we will see in Sections 11 and 12 they constitute a powerful tool for relating different valuation models.

Definition 5. *Given two valuation models* $\mathcal{M}, \mathcal{N}$*, a logical relation* $\mathscr{R}$ *between* $\mathcal{M}$ *and* $\mathcal{N}$ *is a family* $\{\mathscr{R}_A\}_{A \in \mathbb{T}^0}$ *of binary relations* $\mathscr{R}_A \subseteq \mathcal{M}_A \times \mathcal{N}_A$ *such that for all* $A, B \in \mathbb{T}^0$*,* $f \in \mathcal{M}_{A \to B}$ *and* $g \in \mathcal{N}_{A \to B}$ *we have:*

$$f \ \mathscr{R}_{A \to B} \ g \text{ iff } \forall h \in \mathcal{M}_A, h' \in \mathcal{N}_A [h \ \mathscr{R}_A \ h' \Rightarrow f(h) \ \mathscr{R}_B \ g(h')].$$

Given $f \in \mathcal{M}_A$ we define $\mathscr{R}_A(f) = \{g \in \mathcal{N}_A \mid f\ \mathscr{R}_A\ g\}$ and, for $Y \subseteq \mathcal{M}_A$, $\mathscr{R}_A(Y) = \bigcup_{f\in Y} \mathscr{R}_A(f)$. Similarly, for $g \in \mathcal{N}_A$ and $Z \subseteq \mathcal{N}_A$ we have $\mathscr{R}_A^-(g) = \{f \in \mathcal{N}_A \mid f\ \mathscr{R}_A\ g\}$ and $\mathscr{R}_A^-(Z) = \bigcup_{g\in Z} \mathscr{R}_A^-(g)$.

It is well known that a logical relation $\mathscr{R}$ is univocally determined by the value of $\mathscr{R}_0$, and that the fundamental lemma of logical relations holds [2].

Lemma 1 (Fundamental Lemma). *Let $\mathscr{R}$ be a logical relation between $\mathcal{M}$ and $\mathcal{N}$ then, for all closed M having simple type A, we have $[\![M]\!]^{\mathcal{M}}\ \mathscr{R}_A\ [\![M]\!]^{\mathcal{N}}$.*

9 Uniform Intersection Types and CDV$^\omega$

A useful approach to prove that a general decision problem is undecidable, is to identify a "sufficiently difficult" fragment of the problem. For instance, Urzyczyn in [10] shows the undecidability of inhabitation for a proper subset $\mathcal{G}$ of intersection types called *game types* in [3]. Formally, $\mathcal{G} = \mathbb{A} \cup \mathcal{B} \cup \mathcal{C}$ where:

$$\mathcal{A} = \mathbb{A}^\wedge, \mathcal{B} = (\mathcal{A} \to \mathcal{A})^\wedge, \mathcal{C} = (\mathcal{D} \to \mathcal{A})^\wedge \text{ for } \mathcal{D} = \{\sigma \wedge \tau \mid \sigma, \tau \in (\mathcal{B} \to \mathcal{A})\}.$$

(Recall that the notations $Y^\wedge$ and $Y \to Z$ were introduced in Subsection 8.1.) In our case we focus on intersection types that are *uniform* with simple types, in the sense that such intersection types follow the structure of the simple types.

Let us fix an arbitrary set $X \subseteq \mathbb{A}$. We write $\mathbb{T}_\wedge^X$ for the set of intersection types based on X.

Definition 6. *The set $\Xi_X(A)$ of* intersection types uniform with $A \in \mathbb{T}^0$ *is defined by induction on A as follows:*

$$\Xi_X(0) = X^\wedge, \qquad \Xi_X(B \to C) = (\Xi_X(B) \to \Xi_X(C))^\wedge.$$

When there is little danger of confusion, we simply write $\Xi(A)$ for $\Xi_X(A)$.

It turns out that game types are all uniform: $\mathcal{A} \subseteq \Xi_{\mathbb{A}}(0)$, $\mathcal{B} \subseteq \Xi_{\mathbb{A}}(0 \to 0)$ and $\mathcal{D} \subseteq \Xi_{\mathbb{A}}((0 \to 0) \to 0)$ thus $\mathcal{C} \subseteq \Xi_{\mathbb{A}}(((0 \to 0) \to 0) \to 0)$. Therefore the inhabitation problem for uniform intersection types over $\mathbb{A}$ is undecidable too.

Theorem 15 (Urzyczyn revisited). *The problem of deciding whether a type $\sigma \in \bigcup_{A\in\mathbb{T}^0, X\subseteq_{\mathrm{f}}\mathbb{A}} \Xi_X(A)$ is inhabited in CDV is undecidable.*

For technical reasons, that will be clarified in the next section, we need to introduce the system CDV$^\omega$ over $\mathbb{A} \cup \{\omega\}$, a variation of CDV where intersection types are extended by adding a distinguished element ω at ground level.

In this framework, the set $\Xi_{X\cup\{\omega\}}(A)$ of *intersection types with ω uniform with A* will be denoted by $\Xi_X^\omega(A)$, or just $\Xi^\omega(A)$ when X is clear. We write ω_A for the type in $\Xi^\omega(A)$ defined by $\omega_0 = \omega$ and $\omega_{B\to C} = \omega_B \to \omega_C$.

The system CDV$^\omega$ over $\mathbb{T}_\wedge^{\mathbb{A}\cup\{\omega\}}$, whose judgments are denoted by $\Gamma \vdash_\wedge^\omega M : \sigma$, is generated by adding the following rule to the definition of $\leq$ in Figure 1(b):

$$\frac{\sigma \in \Xi_{\mathbb{A}}^\omega(A)}{\sigma \leq \omega_A}\ (\leq_A)$$

Therefore CDV$^\omega$ is different from the usual intersection type systems with ω. By construction, for every $A \in \mathbb{T}^0$, the type ω_A is a maximal element of $\Xi^\omega(A)$. Using [3, Thm. 14A.7], we easily get that the Inversion Lemma (Theorem 14) still works for CDV$^\omega$, while the β-soundness holds in the following restricted form.

Recall that $\simeq$ stands for the equivalence generated by $\leq$.

Theorem 16 (β-soundness for CDV$^\omega$). *Let $k \geq 1$. Suppose $\gamma_1 \to \gamma_2 \not\simeq \omega_A$ for all $A \in \mathbb{T}^0$ and $\bigwedge_{i=1}^k \sigma_i \to \rho_i \leq \gamma_1 \to \gamma_2$, then there is a non-empty subset $K \subseteq [1;k]$ such that $\gamma_1 \leq \bigwedge_{i\in K} \sigma_i$ and $\bigwedge_{i\in K} \rho_i \leq \gamma_2$.*

We now provide some useful properties of uniform intersection types.

Lemma 2. *Let $\sigma \in \Xi^\omega(A)$ and $\tau \in \Xi^\omega(A')$. Then we have that $\sigma \leq \tau$ entails $A = A'$.*

To distinguish arbitrary contexts from contexts containing uniform intersection types (with or without ω) we introduce some terminology.

We say that a context Γ is a *Ξ-context* (resp. *Ξ^ω-context*) if it ranges over uniform intersection types (resp. with ω). A Ξ-context (resp. Ξ^ω-context) $\Gamma = x_1 : \sigma_1, \ldots, x_n : \sigma_n$ is *uniform with $\Delta = x_1 : A_1, \ldots, x_n : A_n$* if every σ_i belongs to $\Xi(A_i)$ (resp. to $\Xi^\omega(A_i)$).

Lemma 3. *Let $\rho \in \mathbb{T}_\wedge^{\mathbb{A}\cup\{\omega\}}$, $\tau \in \Xi^\omega(B)$ and Γ be a Ξ^ω-context. Then we have that $\Gamma, x : \tau \vdash_\wedge^\omega xN_1 \cdots N_k : \rho$ iff there are $A, A_1, \ldots, A_k \in \mathbb{T}^0$ and $\sigma \in \Xi^\omega(A)$ and $\tau_i \in \Xi^\omega(A_i)$ for i in $[1;k]$ such that $B = A_1 \to \cdots \to A_k \to A$ and:*

1. *$\sigma \leq \rho$,*
2. *$\Gamma, x : \tau \vdash_\wedge^\omega xN_1 \cdots N_k : \sigma$,*
3. *$\tau \leq \tau_1 \to \cdots \to \tau_k \to \sigma$,*
4. *$\Gamma, x : \tau \vdash_\wedge^\omega N_i : \tau_i$ for all i in $[1;k]$.*

Furthermore, if Γ is a Ξ-context, $\rho \in \mathbb{T}_\wedge^{\mathbb{A}}$ and $\tau \in \Xi(B)$, then σ and the τ_i for i in $[1;k]$ may also be chosen as uniform intersection types without ω (while the type judgments $\vdash_\wedge^\omega$ still need to be in CDV$^\omega$*).*

Theorem 17 (Uniform Inversion Lemma for CDV$^\omega$). *Let $\sigma \in \Xi^\omega(A)$ and Γ be a Ξ^ω-context. Then we have that (where we suppose that each term in a type judgment is in normal form):*

1. *$\Gamma \vdash_\wedge^\omega x : \sigma$ iff $\Gamma(x) \leq \sigma$,*
2. *$\Gamma \vdash_\wedge^\omega MN : \sigma$ iff there exist $B \in \mathbb{T}^0$ and $\tau \in \Xi^\omega(B)$ such that $\Gamma \vdash_\wedge^\omega M : \tau \to \sigma$ and $\Gamma \vdash_\wedge^\omega N : \tau$,*
3. *$\Gamma \vdash_\wedge^\omega \lambda x.N : \sigma$ iff $A = B \to C$ and there are $\tau_i \in \Xi^\omega(B), \tau_i' \in \Xi^\omega(C)$ such that $\sigma = \bigwedge_{i=1}^n \tau_i \to \tau_i'$ and $\Gamma, x : \tau_i \vdash_\wedge^\omega N : \tau_i'$ for all i in $[1;n]$.*

Corollary 1. *For M a normal λ-term, $\sigma \in \Xi^\omega(A)$ and Γ a Ξ^ω-context uniform with Δ, we have that $\Gamma \vdash_\wedge^\omega M : \sigma$ entails $\Delta \vdash M : A$.*

Proof. A simple consequence of the Uniform Inversion Lemma (with Lemma 2 when M is a variable). □

The corollary above does not generalize to arbitrary λ-terms as the following example illustrates. Let $M = \lambda zy.y$ and $N = \lambda x.xx$, then we have that $\vdash^{\omega}_{\wedge} MN : \alpha \to \alpha \in \Xi^{\omega}(0 \to 0)$ since $\vdash^{\omega}_{\wedge} N : \gamma$ and $\vdash^{\omega}_{\wedge} M : \gamma \to \alpha \to \alpha$ where $\gamma = (\beta \wedge (\beta \to \beta)) \to \beta$. However N is not simply typable, hence neither is MN. Note that, while we consider only uniform intersection types, we do not restrict the intersection type systems so that the type γ still may be used in a deduction.

CDV and CDV$^{\omega}$ are equivalent on normal forms in the following sense.

Lemma 4. *For every normal $M \in \Lambda$, for every Ξ-context $\Gamma = x_1 : \tau_1, \ldots, x_n : \tau_n$ uniform with $\Delta = x_1 : A_1, \ldots, x_n : A_n$, and for every $\sigma \in \Xi(A)$ we have:*

$$\Gamma \vdash_{\wedge} M : \sigma \Leftrightarrow \Gamma \vdash^{\omega}_{\wedge} M : \sigma.$$

Proof. ($\Rightarrow$) Trivial, as CDV is a subsystem of CDV$^{\omega}$.

($\Leftarrow$) We proceed by induction on the structure of M. The cases where M is a variable or a λ-abstraction can be treated thanks to Theorem 14 for CDV$^{\omega}$and the induction hypothesis. Concerning the case where $M = x_i N_1 \cdots N_k$, from the ω-free version of Lemma 3, we have that $A_i = B_1 \to \cdots \to B_k \to A$, there exist $\tau_1, \ldots, \tau_k$ respectively in $\Xi(B_1), \ldots, \Xi(B_k)$ such that $\tau_i \leq \tau_1 \to \cdots \to \tau_k \to \sigma$ and $\Gamma \vdash^{\omega}_{\wedge} N_i : \tau_i$ for each i in $[1; k]$. Therefore, by the induction hypothesis, we have that for every i in $[1; k]$, $\Gamma \vdash_{\wedge} N_i : \tau_i$ which entails that $\Gamma \vdash_{\wedge} M : \sigma$. □

10 The Continuous Model over $\mathcal{P}(X)$

Hereafter we consider fixed an arbitrary set $X \subseteq_{\mathrm{f}} \mathbb{A}$. We are going to represent uniform intersection types based on $X \cup \{\omega\}$, as elements of the continuous model $\mathcal{S}$ over $\mathcal{P}(X)$, ordered by set-theoretical inclusion.

Let $\mathcal{S} = \{(\mathcal{S}_A, \sqsubseteq_A)\}_{A \in \mathbb{T}^0} = \mathrm{Cont}(\mathcal{P}(X), \subseteq)$. Each $\mathcal{S}_A$ is a finite join-semilattice and thus a complete lattice. We denote the join by $\sqcup$ and the bottom by $\bot_A$.

Given $f \in \mathcal{S}_A, g \in \mathcal{S}_B$ we write $f \mapsto g$ for the corresponding *step function*:

$$(f \mapsto g)(h) = \begin{cases} g & \text{if } f \sqsubseteq_A h, \\ \bot_B & \text{otherwise.} \end{cases}$$

For all A we define a function $\iota_A : \Xi^{\omega}(A) \to \mathcal{S}_A$ by induction on A as follows.

Definition 7. *For $\alpha \in X$ and $\sigma, \tau \in \Xi^{\omega}(0)$ we let $\iota_0(\alpha) = \{\alpha\}$, $\iota_0(\omega) = \bot_0 = \emptyset$, $\iota_0(\sigma \wedge \tau) = \iota_0(\sigma) \sqcup \iota_0(\tau)$. For $\sigma, \tau \in \Xi^{\omega}(A \to B)$ we define:*

$$\iota_{A \to B}(\sigma \to \tau) = \iota_A(\sigma) \mapsto \iota_B(\tau), \qquad \iota_{A \to B}(\sigma \wedge \tau) = \iota_{A \to B}(\sigma) \sqcup \iota_{A \to B}(\tau).$$

Remark 1. Given $\sigma \in \Xi^{\omega}(A)$, we have that $\sigma \simeq \omega_A$ entails $\iota_A(\sigma) = \bot_A$.

Thanks to the presence of the maximal element ω_A, the correspondence between $\Xi^{\omega}(A)$ and $\mathcal{S}_A$ is actually very faithful (in the sense of Corollary 2).

Lemma 5. *Let $h = \bigsqcup_{i=1}^{n} f_i \mapsto g_i$, then for every f we have:*

(i) $h(f) = \bigsqcup_{i \in K} g_i$ *where* $K = \{i \in [1;n] \mid f_i \sqsubseteq f\}$.
(ii) $h \sqsubseteq f$ *iff* $g_i \sqsubseteq f(f_i)$ *for all* $1 \leq i \leq n$.

Lemma 6. *Step functions are generators:* $\forall f \in \mathcal{S}_{A \to B}$, $f = \bigsqcup_{g \in \mathcal{S}_A} g \mapsto f(g)$.

Proof. Let $h = \bigsqcup_{g \in \mathcal{S}_A} g \mapsto f(g)$. We need to prove that, for every $g \in \mathcal{S}_A$, $f(g) = h(g)$. From Lemma 5(i), we have that $h(g) = \bigsqcup_{g' \sqsubseteq g} f(g')$. Since f is monotone, we have that for every $g' \sqsubseteq g$, $f(g') \sqsubseteq f(g)$ and therefore $\bigsqcup_{g' \sqsubseteq g} f(g') \sqsubseteq f(g)$. Since obviously $f(g) \sqsubseteq \bigsqcup_{g' \sqsubseteq g} f(g')$, we obtain $f(g) = \bigsqcup_{g' \sqsubseteq g} f(g') = h(g)$. □

Lemma 7. *For all $A \in \mathbb{T}^0$, $\sigma, \tau \in \Xi^\omega(A)$ we have $\sigma \leq \tau$ iff $\iota_A(\tau) \sqsubseteq \iota_A(\sigma)$.*

Proof. We proceed by induction on A. In case $A = 0$, the equivalence is clear since $\mathcal{P}(X)$ is the free $\sqcup$-semilattice with bottom over X and $\Xi^\omega(0)/\simeq$ is the free $\wedge$-semilattice with top over X.

In case $A = B \to C$, we have two subcases. Case 1, $\tau \simeq \omega_D$ for some $D \in \mathbb{T}^0$. Then by Lemma 2 we get $D = A$, by Remark 1 we get $\iota_A(\tau) = \bot_A$ and the equivalence follows since both $\sigma \leq \tau$ and $\iota_A(\tau) \sqsubseteq \iota_A(\sigma)$ hold. Case 2, $\sigma = \bigwedge_{i=1}^{n} \sigma_i \to \sigma'_i$, $\tau = \bigwedge_{j=1}^{m} \tau_j \to \tau'_j$ and $\tau \not\simeq \omega_D$ for any $D \in \mathbb{T}^0$. By Remark 1 we can assume, without loss of generality, that for every j in $[1;m]$ we have $\tau_j \to \tau'_j \not\simeq \omega_D$ for all $D \in \mathbb{T}^0$. (Indeed for those k such that $\tau_k \to \tau'_k \simeq \omega_D$ one reasons as in Case 1.) We now prove the equivalence for this case.

($\Rightarrow$) If $\sigma \leq \tau$, then by β-soundness, for every j in $[1;m]$, there is K_j included in $[1;n]$ such that $\tau_j \leq \bigwedge_{i \in K_j} \sigma_i$ and $\bigwedge_{i \in K_j} \sigma'_i \leq \tau'_j$. By the induction hypothesis:

$$(1) \quad \bigsqcup_{i \in K_j} \iota_B(\sigma_i) \sqsubseteq \iota_B(\tau_j) \qquad (2) \quad \iota_C(\tau'_j) \sqsubseteq \bigsqcup_{i \in K_j} \iota_C(\sigma'_i)$$

We now prove that, for every $f \in \mathcal{S}_B$, $\iota_A(\tau)(f) \sqsubseteq \iota_A(\sigma)(f)$. From Lemma 5(i), we get $\iota_A(\tau)(f) = \bigsqcup_{j \in J} \iota_C(\tau'_j)$ where $J = \{j \in [1;m] \mid \iota_B(\tau_j) \sqsubseteq f\}$. By definition of J, we have that $\bigsqcup_{j \in J} \iota_B(\tau_j) \sqsubseteq f$ so, by (1), we obtain $\bigsqcup_{j \in J, i \in K_j} \iota_B(\sigma_i) \sqsubseteq f$. Therefore by Lemma 5(i), we get $\bigsqcup_{j \in J, i \in K_j} \iota_C(\sigma'_i) \sqsubseteq \iota_A(\sigma)(f)$ and, using (2), we obtain $\iota_A(\tau)(f) \sqsubseteq \iota_A(\sigma)(f)$. As a conclusion we have $\iota_A(\tau) \sqsubseteq \iota_A(\sigma)$.

($\Leftarrow$) If $\iota_A(\tau) \sqsubseteq \iota_A(\sigma)$, then we have in particular $\iota_A(\tau)(\iota_B(\tau_j)) \sqsubseteq \iota_A(\sigma)(\iota_B(\tau_j))$ for each $j \in [1,m]$. From Lemma 5(i), we have that $\iota_A(\tau)(\iota_B(\tau_j)) = \bigsqcup_{i \in I_j} \iota_C(\tau'_i)$ where $I_j = \{i \in [1;m] \mid \tau_i \leq \tau_j\}$. Since $\tau_j \leq \tau_j$ we must have $j \in I_j$ and therefore, we obtain $\iota_C(\tau'_j) \sqsubseteq \iota_A(\tau)(\iota_B(\tau_j))$. So, again by Lemma 5(i), we have that $\iota_A(\sigma)(\iota_B(\tau_j)) = \bigsqcup_{k \in K_j} \iota_C(\sigma'_k)$ where $K_j = \{k \in [1;n] \mid \tau_j \leq \sigma_k\}$. Thus we get $\iota_C(\tau'_j) \sqsubseteq \bigsqcup_{k \in K_j} \iota_C(\sigma'_k)$ and hence, by the induction hypothesis, $\bigwedge_{k \in K_j} \sigma'_k \leq \tau'_j$. Now, by definition of K_j, we also have $\tau_j \leq \bigwedge_{k \in K_j} \sigma_k$. As we can find such a K_j for every j in $[1;m]$, we can finally conclude that $\sigma \leq \tau$. □

Corollary 2. *The map ι_A is an order-reversing bijection on $\Xi^\omega(A)/\simeq$.*

Proof. If $\tau \leq \sigma$ and $\sigma \leq \tau$, then Lemma 7 implies that $\iota_A(\tau) = \iota_A(\sigma)$. From this it ensues that ι_A is an order-reversing injection. To prove that it is actually a bijection, we need to show that ι_A is surjective. We proceed by induction on A. Clearly when $A = 0$, ι_A is surjective. If $A = B \to C$ then we get from the induction hypothesis that ι_B and ι_C are bijections between $\Xi^\omega(B)/\simeq$ and $\mathcal{S}_B$, and between $\Xi^\omega(C)/\simeq$ and $\mathcal{S}_C$, respectively. Now, given f in $\mathcal{S}_A$, we define $\tau_f \in \Xi^\omega(A)$ to be $\bigwedge_{g\in\mathcal{S}_B} \iota_B^{-1}(g) \to \iota_C^{-1}(f(g))$. But, $\iota_{A\to B}(\tau_f) = \bigsqcup_{g\in\mathcal{S}_B} g \mapsto f(g)$ which is equal to f by Lemma 6. □

The above results are related to Stone duality for intersection types (cf. [1]).

Proposition 1. *Let M be a normal term such that $x_1 : A_1, \ldots, x_n : A_n \vdash M : A$. Then for all $\tau_i \in \Xi^\omega(A_i)$, $\sigma \in \Xi^\omega(A)$ the following two sentences are equivalent:*

1. $x_1 : \tau_1, \ldots, x_n : \tau_n \vdash^\omega_\wedge M : \sigma$,
2. $\iota_A(\sigma) \sqsubseteq [\![M]\!]^{\mathcal{S}}_\nu$, *for all valuations ν such that $\nu(x_i) = \iota_{A_i}(\tau_i)$.*

Proof. Let $\Delta = x_1 : A_1, \ldots, x_n : A_n$ and $\Gamma = x_1 : \tau_1, \ldots, x_n : \tau_n$.
$(1 \Rightarrow 2)$ We proceed by structural induction on M.

- In case $M = x_i$, then $\tau_i \leq \sigma$ and, by Lemma 7, $\iota_{A_i}(\sigma) \sqsubseteq \iota_{A_i}(\tau_i) = [\![x_i]\!]_\nu$.
- In case $M = NP$, then, from Theorem 17(2), there are $B \in \mathbb{T}^0$ and $\tau \in \Xi^\omega(B)$ such that $\Gamma \vdash^\omega_\wedge N : \tau \to \sigma$ and $\Gamma \vdash^\omega_\wedge P : \tau$. By induction $\iota_{B\to A}(\tau \to \sigma) \sqsubseteq [\![N]\!]_\nu$ and $\iota_B(\tau) \sqsubseteq [\![P]\!]_\nu$, thus, $\iota_A(\sigma) = \iota_{B\to A}(\tau \to \sigma)(\iota_B(\tau)) \sqsubseteq [\![N]\!]_\nu(\iota_B(\tau))$ and, by monotonicity, $[\![N]\!]_\nu(\iota_B(\tau)) \sqsubseteq [\![N]\!]_\nu([\![P]\!]_\nu) = [\![NP]\!]_\nu$. From this we finally get $\iota_A(\sigma) \sqsubseteq [\![NP]\!]_\nu$.
- In case $M = \lambda x.N$, then by Theorem 17(3) we have that $A = B \to C$ and, for all $j \in [1; n]$, there are $\sigma_j \in \Xi^\omega(B)$, $\sigma'_j \in \Xi^\omega(C)$ such that $\sigma = \bigwedge_{j=1}^n \sigma_j \to \sigma'_j$ and $\Gamma, x : \sigma_j \vdash^\omega_\wedge N : \sigma'_j$. Thus, by induction hypothesis, we get $\iota_C(\sigma'_j) \sqsubseteq [\![N]\!]_{\nu[x:=\iota_B(\sigma_j)]}$. From Lemma 5(ii) it ensues that $\iota_A(\sigma) \sqsubseteq [\![M]\!]_\nu$.

$(2 \Rightarrow 1)$ It suffices to establish by induction that $[\![M]\!]_\nu = \iota_A(\sigma)$, for all ν such that $\nu(x_i) = \iota_{A_i}(\tau_i)$, entails $\Gamma \vdash^\omega_\wedge M : \sigma$. Indeed, if τ is such that $\iota_A(\tau) \sqsubseteq [\![M]\!]_\nu$ then by Lemma 7 and $\sigma \leq \tau$ we obtain, using the subsumption rule, that $\Gamma \vdash^\omega_\wedge M : \tau$.

- If $M = x_i$, then $[\![x_i]\!]_\nu = \iota_{A_i}(\tau_i) = \iota_A(\sigma)$ and $\sigma \simeq \tau_i$. Thus $\Gamma \vdash^\omega_\wedge x_i : \sigma$.
- If $M = NP$, then there is B such that $\Delta \vdash N : B \to A$ and $\Delta \vdash P : B$. By Corollary 2, there are $\tau \in \Xi^\omega(B \to A), \rho \in \Xi^\omega(B)$ such that $[\![N]\!]_\nu = \iota_{B\to A}(\tau)$ and $[\![P]\!]_\nu = \iota_B(\rho)$. The induction hypothesis implies that $\Gamma \vdash^\omega_\wedge N : \tau$ and $\Gamma \vdash^\omega_\wedge P : \rho$ are derivable. By hypothesis we know that $[\![M]\!]_\nu = \iota_A(\sigma)$. From Lemma 5(ii), since $\iota_A(\sigma) = [\![M]\!]_\nu = [\![N]\!]_\nu([\![P]\!]_\nu) = \iota_{B\to A}(\tau)(\iota_B(\rho))$, we have $\iota_B(\rho) \mapsto \iota_A(\sigma) \sqsubseteq \iota_{B\to A}(\tau)$ and thus, by Lemma 7, $\tau \leq \rho \to \sigma$. Hence $\Gamma \vdash^\omega_\wedge N : \rho \to \sigma$ is derivable, which implies that $\Gamma \vdash^\omega_\wedge M : \sigma$ is derivable.
- If $M = \lambda x.N$, then $A = B \to C$. By Corollary 2 we can choose, for every $g \in \mathcal{S}_B$, $\sigma_g \in \Xi^\omega(B)$ such that $\iota_B(\sigma_g) = g$ and $\tau_g \in \Xi^\omega(C)$ such that $\iota_C(\tau_g) = [\![N]\!]_{\nu[x:=g]} = [\![M]\!]_\nu(g)$. By the induction hypothesis, for every $g \in \mathcal{S}_B$, we have $\Gamma, x : \sigma_g \vdash^\omega_\wedge N : \tau_g$. Therefore, $\Gamma \vdash^\omega_\wedge M : \sigma_g \to \tau_g$ and $\Gamma \vdash^\omega_\wedge M : \bigwedge_{g\in\mathcal{S}_B} \sigma_g \to \tau_g$. By definition $\iota_A(\bigwedge_{g\in\mathcal{S}_B} \sigma_g \to \tau_g) = \bigsqcup_{g\in\mathcal{S}_b} \iota_B(\sigma_g) \mapsto \iota_C(\tau_g)$ $= \bigsqcup_{g\in\mathcal{S}_b} g \mapsto [\![M]\!]_\nu(g)$ which is equal, by Lemma 6, to $[\![M]\!]_\nu$. □

11 Inhabitation Reduces to Definability

We now prove that the undecidability of the Definability Problem follows from the undecidability of the inhabitation problem (for game types) in CDV. A preliminary version of this result was announced in the invited paper [8].

The proof we present here is obtained by linking via a suitable logical relation $\mathscr{I}$ the continuous model $\mathcal{S}$ built in the previous section and $\mathcal{F} = \{\mathcal{F}_A\}_{A\in\mathbb{T}^0} = \mathrm{Full}(\mathcal{P}(X))$, where $X \subseteq_{\mathrm{f}} \mathbb{A}$. Let $\mathscr{I}$ be the logical relation between $\mathcal{S}$ and $\mathcal{F}$ generated by taking the identity at ground level (indeed $\mathcal{S}_0 = \mathcal{F}_0 = \mathcal{P}(X)$).

Lemma 8. *$\mathscr{I}$ is a* logical retract*, i.e. at every level $A \in \mathbb{T}^0$ we have $\forall f_1, f_2 \in \mathcal{S}_A$, $\mathscr{I}_A(f_1) \cap \mathscr{I}_A(f_2) \neq \emptyset$ iff $f_1 = f_2$. Equivalently, both next statements hold:*

(i) for all $f \in \mathcal{S}_A$ there is $g \in \mathcal{F}_A$ such that $f\ \mathscr{I}_A\ g$,
(ii) for all $f, f' \in \mathcal{S}_A, g \in \mathcal{F}_A$ if $f\ \mathscr{I}_A\ g$ and $f'\ \mathscr{I}_A\ g$ then $f = f'$.

Proof. We prove the main statement by induction on A, then both items follow. The base case $A = 0$ is trivial, so we consider the case $A = B \to C$.

($\Rightarrow$) By definition of $\mathscr{I}_A(f_1), \mathscr{I}_A(f_2)$ we have:

$$\mathscr{I}_A(f_1) \cap \mathscr{I}_A(f_2) = \{h \mid \forall g \in \mathcal{S}_B, \forall k \in \mathscr{I}_B(g), h(k) \in \mathscr{I}_C(f_1(g)) \cap \mathscr{I}_C(f_2(g))\}.$$

Now, $\mathscr{I}_A(f_1) \cap \mathscr{I}_A(f_2) \neq \emptyset$ entails $\mathscr{I}_C(f_1(g)) \cap \mathscr{I}_C(f_2(g)) \neq \emptyset$ for all $g \in \mathcal{S}_B$. By induction, this holds when $f_1(g) = f_2(g)$ for all $g \in \mathcal{S}_B$, i.e. when $f_1 = f_2$.

($\Leftarrow$) If $f_1 = f_2$ then $\mathscr{I}_A(f_1) = \{h \mid \forall g \in \mathcal{S}_B, \forall k \in \mathscr{I}_B(g), h(k) \in \mathscr{I}_C(f_1(g))\}$. To prove $\mathscr{I}_A(f_1) \neq \emptyset$, we build a relation $h \subseteq \mathcal{F}_B \times \mathcal{F}_C$ that is actually functional and belongs to it. Fix any $d \in \mathcal{F}_C$ and, for every $g \in \mathcal{S}_B$, an element $r_g \in \mathscr{I}_C(f_1(g))$ which exists by induction hypothesis. Define h as the smallest relation such that $(k, r_g) \in h$ if $k \in \mathscr{I}_B(g)$, and $(k, d) \in h$ if $k \notin \bigcup_{g\in\mathcal{S}_B} \mathscr{I}_B(g)$. As, by induction hypothesis, $\mathscr{I}_B(g_1)$ and $\mathscr{I}_B(g_2)$ are disjoint for all $g_1 \neq g_2$ then h is functional. By construction, $h \in \mathscr{I}_C(f_1(g))$. □

As a consequence we get, for every subset $S \subseteq \mathcal{S}_A$, that $\mathscr{I}_A^-(\mathscr{I}_A(S)) = S$. Given $f \in \mathcal{S}_A$ we write $f{\uparrow}$ for its upward closure in $\mathcal{S}_A$: $\{f' \in \mathcal{S}_A \mid f \sqsubseteq f'\}$.

Proposition 2. *Let $\sigma \in \Xi(A)$. For every normal λ-term M having type A we have $\vdash_\wedge M : \sigma$ iff $[\![M]\!]^{\mathcal{F}} \in \mathscr{I}_A(\iota_A(\sigma){\uparrow})$.*

Proof. We have the following computable chain of equivalences:

$$\begin{aligned}
\vdash_\wedge M : \sigma &\Leftrightarrow \vdash_\wedge^\omega M : \sigma, && \text{by Lemma 4,}\\
&\Leftrightarrow [\![M]\!]^{\mathcal{S}} \in \iota_A(\sigma){\uparrow}, && \text{by Proposition 1,}\\
&\Leftrightarrow [\![M]\!]^{\mathcal{F}} \in \mathscr{I}_A(\iota_A(\sigma){\uparrow}), && \text{by Lemma 1 plus Lemma 8.}
\end{aligned}$$ □

Theorem 18. *The undecidability of the Definability Problem follows by a reduction from the one of the Inhabitation Problem for game types, Theorem 11(2).*

Proof. Suppose by contradiction that DP is decidable. We want to decide whether $\sigma \in \bigcup_{A \in \mathbb{T}^0, X \subseteq_f \mathbb{A}} \Xi_X(A)$ is inhabited in CDV. By Theorem 12 and Corollary 1 we can focus on normal simply typed λ-terms. Now we can take the set Y of all atoms in σ, compute the simple type A such that $\sigma \in \Xi_Y(A)$, and effectively construct the finite set $\mathscr{I}_A(\iota_A(\sigma)\uparrow) \subseteq \mathrm{Full}(Y)$. If DP is decidable, then we can also decide with finitely many tests whether there is a λ-definable $f \in \mathscr{I}_A(\iota_A(\sigma)\uparrow)$. By Proposition 2 such an f exists if and only if σ is inhabited. This yields a reduction of IHP for game types (hence for uniform types, Theorem 15) to DP. □

12 Definability Reduces to Inhabitation

In this section we prove the converse of Theorem 18, namely that the undecidability of inhabitation follows directly from the undecidability of λ-definability in the full model $\mathcal{F} = \mathrm{Full}(X)$ over a fixed set $X \subseteq_f \mathbb{A}$. The main idea is a simple embedding of the elements of $\mathcal{F}$ into the uniform intersection types.

Also in this proof the continuous model $\mathcal{S} = \mathrm{Cont}(\mathcal{P}(X), \subseteq)$ will play a key role. (Remark that the ground set of $\mathcal{S}$ is still $\mathcal{P}(X)$, while $\mathcal{F}$ is now over X.) We start by defining an injection $\varphi_A : \mathcal{F}_A \to \mathcal{S}_A$ by induction on A:

- if $A = 0$, then $\varphi_A(f) = \{f\}$,
- if $A = B \to C$, then $\varphi_A(f) = \bigsqcup_{g \in \mathcal{F}_B} \varphi_B(g) \mapsto \varphi_C(f(g))$.

Now, given f in $\mathcal{F}_A$ we define an intersection type ξ_f in $\Xi(A)$ as follows:

- if $A = 0$, then $\xi_f = f$,
- if $A = B \to C$, then $\xi_f = \bigwedge_{g \in \mathcal{F}_B} \xi_g \to \xi_{f(g)}$.

Lemma 9. *For every f in $\mathcal{F}_A$, we have $\varphi_A(f) = \iota_A(\xi_f)$.*

We consider the logical relation $\mathscr{J}$ between the full model $\mathcal{F}$ and the continuous model $\mathcal{S}$ generated by $\mathscr{J}_0 = \{(f, F) \mid f \in F \subseteq \mathcal{F}_0\}$.

Lemma 10. *For every $f \in \mathcal{F}_A$ and $g \in \mathcal{S}_A$ we have $f \mathscr{J}_A g$ iff $\varphi_A(f) \sqsubseteq g$.*

Proof. By induction on A, the case $A = 0$ being obvious. Let $A = B \to C$.

($\Rightarrow$) Suppose $f \mathscr{J}_A g$. We want to prove that $\varphi_A(f) \sqsubseteq g$. That is, for all $h \in \mathcal{S}_B$, we have $\varphi_A(f)(h) \sqsubseteq g(h)$. Let $h \in \mathcal{S}_B$, then by definition of φ_A, we have $\varphi_A(f)(h) = \bigsqcup\{\varphi_C(f(k)) \mid \varphi_B(k) \sqsubseteq h, k \in \mathcal{F}_B\}$. But $\varphi_B(k) \sqsubseteq h$ implies $k \mathscr{J}_B h$ by induction hypothesis, which implies that $f(k) \mathscr{J}_C g(h)$ since $f \mathscr{J}_A g$. Now using the induction hypothesis for C, we get $\varphi_C(f(k)) \sqsubseteq g(h)$. That is, $\varphi_A(f)(h)$ is a supremum of things all of which are below $g(h)$, thus $\varphi_A(f)(h) \sqsubseteq g(h)$.

($\Leftarrow$) Suppose $\varphi_A(f) \sqsubseteq g$. Let $h \in \mathcal{F}_B$ and $h' \in \mathcal{S}_B$ with $h \mathscr{J}_B h'$, that is, by the induction hypothesis, with $\varphi_B(h) \sqsubseteq h'$. We want to show that $f(h) \mathscr{J}_C g(h')$ or, equivalently, again by the induction hypothesis, that $\varphi_C(f(h)) \sqsubseteq g(h')$. Now, by definition, $\varphi_A(f)(h') = \bigsqcup\{\varphi_C(f(k)) \mid \varphi_B(k) \sqsubseteq h', k \in \mathcal{F}_B\}$, and by assumption $h \in \mathcal{F}_B$ and $\varphi_B(h) \sqsubseteq h'$, so $\varphi_C(f(h)) \sqsubseteq \varphi_A(f)(h')$. On the other hand, $\varphi_A(f) \sqsubseteq g$ as functions on $\mathcal{S}_A$ and $h' \in \mathcal{S}_B$, so $\varphi_A(f)(h') \sqsubseteq g(h')$. By transitivity of the order we obtain $\varphi_C(f(h)) \sqsubseteq g(h')$ as required. □

Proposition 3. *Given f in $\mathcal{F}_A$, we have $[\![M]\!]^{\mathcal{F}} = f$ iff $\vdash_\wedge M : \xi_f$.*

Proof. We have the following computable chain of equivalences:

$$\begin{aligned}
[\![M]\!]^{\mathcal{F}} = f &\Leftrightarrow f \mathscr{J}_A [\![M]\!]^{\mathcal{S}}, && \text{by Lemma 1,}\\
&\Leftrightarrow \varphi(f) \sqsubseteq [\![M]\!]^{\mathcal{S}}, && \text{by Lemma 10,}\\
&\Leftrightarrow \iota_A(\xi_f) \sqsubseteq [\![M]\!]^{\mathcal{S}}, && \text{by Lemma 9,}\\
&\Leftrightarrow \vdash_\wedge M : \xi_f, && \text{by Proposition 1.}
\end{aligned}$$

□

Therefore f is definable iff ξ_f is inhabited. This yields a reduction of the Definability Problem (resp. DP_n) to the Inhabitation Problem (resp. IHP_n).

Theorem 19. *1. The undecidability of IHP_n for all $n > 1$ follows by a reduction from the undecidability of DP_n for all $n > 1$, Theorem 10(2).*
2. The undecidability of the Inhabitation Problem follows by a reduction from the undecidability of the Definability Problem, Theorem 10(1).

References

1. S. Abramsky. *Domain theory in logical form.* In Symposium on Logic and Computer Science (LICS'87), IEEE Computer Science Press, pp. 47-53, 1987.
2. R. Amadio and P.-L. Curien. *Domains and lambda-calculi.* Cambridge Tracts in Theoretical Computer Science, no. 46, Cambridge University Press, 1998.
3. H.P. Barendregt, W. Dekkers and R. Statman. *Lambda calculus with types.* To appear. Draft available at http://www.cs.ru.nl/∼henk/book.pdf.
4. M. Coppo, M. Dezani-Ciancaglini, and B. Venneri. *Functional characters of solvable terms.* Mathematical Logic Quarterly, Volume 27, Issue 2-6, pages 45-58, 1981.
5. T. Joly. *Encoding of the halting problem into the monster type & applications.* Typed Lambda Calculi and Applications (TLCA'03), LNCS, vol. 2701, pp. 153-166, 2003.
6. R. Loader. *The undecidability of lambda definability.* In Logic, Meaning and Computation: Essays in Memory of Alonzo Church, 331-342, 2001.
7. G. Plotkin. *Lambda definability and logical relations.* Memorandum SAI-RM-4, School of Artificial Intelligence, University of Edinburgh, 1973.
8. S. Salvati. *Recognizability in the simply typed lambda-calculus.* Logic, Language, Information and Computation (WoLLIC'09), LNCS, vol. 5514, pp. 48-60, 2009.
9. R. Statman. *Completeness, invariance and λ-definability.* The Journal of Symbolic Logic, vol. 47, no. 1, pp. 17-26, 1982.
10. P. Urzyczyn. *The emptiness problem for intersection types.* The Journal of Symbolic Logic, vol. 64, no. 3, pp. 1195-1215, 1999.