FREC delivrable 12: Stone duality in semantics vs. Stone duality in variety theory

Université de Bordeaux, INRIA, CNRS, LaBRI UMR5800

Abstract. This report is the 12th delivrable of the FREC project. This report contains the description of a denotational semantics for λY-calculus that captures precisely weak Monadic Second Order Logic on the Böhm trees defined by λY-terms. Exploiting Stone duality we have been able to turn properties expressed by those models into an inference system. Moreover we exploited internal dualities of the model so as to exhibit an original type system that refutes properties of λY-terms.
We give a denotational semantics of λY-calculus that is capable of computing properties expressed in weak monadic second-order logic (WMSO). λY-calculus faithfully models control in higher-order programs. Our model construction allows to reduce verification to evaluation. The model that we propose presents a lot of symmetries that reflect elegantly the dualities of weak alternating automata at all orders. Following the domain in logical form approach, we derive a type system for verifying WMSO properties.

1 Introduction

We present a contribution to semantic based approaches to verification and transformation of higher-order functional programs. We consider the class of programs written in the simply-typed calculus with recursion and finite base types. The semantic of a program represented by a λY-term is a tree reflecting the control flow of the program. The properties to verify are about the structure of these trees. We construct a denotational semantics of λY-calculus that is capable of computing properties expressed in weak monadic second-order logic. This construction not only allows to reduce verification to evaluation in a model, but it also gives novel type systems, and model-guided program transformations.

λY-calculus offers an abstraction of higher-order programs that faithfully represents higher-order control. The result of a computation of a λY-term is a Böhm tree which is a kind of a normal form adapted to potentially non-terminating computations. In this setting all constants are non-interpreted, and the result, i.e., the Böhm tree, describes the order in which the constants are applied. We are interested in properties of these results, like for example that every occurrence of the *open* constant is eventually followed by an occurrence of the *close* constant.

Weak MSO (WMSO) is a restriction of monadic second-order logic where quantification is restricted to range over finite sets. Recall that properties expressible in WMSO are captured by weak alternating tree automata. WMSO

is sufficiently strong to express safety, reachability and many liveness properties. Over paths, that is degenerated trees where every node has one successor, WMSO is equivalent to full MSO.

The technical question we address here is to determine whether a Böhm tree of a given λY-term is accepted by a given weak alternating automaton. This problem is known to be decidable thanks to the result of Ong [1], but here we present a denotational approach. Our two main contributions are:

- A construction of a finitary model for a given weak alternating automaton. The value of a term in this model determines if the Böhm tree of the term is accepted by the automaton. So model-checking is reduced to evaluation.
- Two typing systems. A typing system deriving statements of the form "the value of a term M is bigger than an element d of the model"; and a typing system for dual properties. These typing systems are based on step functions and co-step functions in the model and follow the methodology coined as *Domains in Logical Form* [2].

Having a model has several advantages over having just a decision procedure. First, it makes verification compositional: the result for a term is calculated from the results for its subterms. In particular, it opens possibilities for a modular approach to the verification of large programs. Next, it allows semantic based program transformations as for example reflection of a given property in a given term [3,4,5]. It also implies a transfer theorem for WMSO [6] with a number of consequences offered by this theorem. It permits to obtain type systems. Finally, models open a way to novel verification algorithms be it through evaluation, type system, or through hybrid algorithms using typing and evaluation at the same time [7].

Historically, Muller and Schupp [8] have proved that the MSO theory of trees generated by pushdown automata is decidable. Knapik, Niwinski, and Urzyczyn [9] have established the relation between higher-order pushdown automata and λY-calculus by showing that Böhm trees of safe λY-terms are precisely trees generated by such automata. They have also established the decidability of the MSO theory of such trees. Ong [1] has shown the decidability of the MSO theory of Böhm trees for all λY-terms. This last result has been revisited in several different ways. Yet none of them is compositional; they take a term of the base type, and unroll it to some infinite object: tree with pointers [1], computation of a higher-order pushdown automaton with collapse [10], a collection of typing judgments that are used to define a game [11], a computation of a Krivine machine [12].

Already some time ago, Aeligh [13] has discovered that for properties expressed by tree automata with trivial acceptance conditions (TAC automata) there is a much easier method to solve the model checking problem. The core of his approach can be formulated by saying that the model-checking problem for such properties can be reduced to evaluation in a specially constructed and simple model. Building on this observation Kobayashi proposed a type system for such properties and constructed a tool based on it [14]. This in turn opened a

way to an active ongoing research resulting in the steady improvement of the capacities of the model-checking tools. Accordingly, our model and typing systems set the stage for practical verification of WMSO properties.

The model approach to verification of λY-calculus is quite recent. In [4] we have shown that simple models with greatest fixpoints capture exactly properties expressed with TAC automata. We have then extended these models to permit the detection of divergence. The model approach reveals how to capture properties of Böhm trees and then extend them to invariants of computation. Moreover techniques used to construct models and prove their correctness rely on usual techniques of domain theory [15], offering an alternative point of view to techniques based on unrolling. The simplicity offered by models is exemplified by Haddad's [5] recent work giving simple compositional and semantic based transformations of λY-terms.

Most existing model-checking tools handle only TAC automata. This is due the simplicity of Kobayashi's typing system mentioned above. Kobayashi and Ong [11] have proposed a type based approach to model-checking of all MSO properties. It consists of two steps: (i) a typing system is used to construct judgments about terms without fixpoints, (ii) these judgments are used to construct a parity game that is solved to determine if a property holds or not. So the semantics of fixpoints is given by games. Very recently [16], a new very promising model-checking algorithm using this approach has been proposed. It deals with both TAC automata and their duals, and exploits the duality between two players in the parity game. It is likely that this idea can be adapted to our models since in our case fixpoints and the duality are present directly in the model.

In the next section we introduce the main objects of our study. Section 3 presents the model construction and proves some of its basic properties. We then show our main theorem (Theorem 3) saying that the models indeed capture weak alternating automata. Section 4 describes the two type systems and proves their soundness and completeness (Theorem 4). In the conclusion section we mention other applications of our model. All the proofs are presented in the appendix.

2 Preliminaries

We quickly fix notations and notions related to simply typed λY-calculus and to Böhm trees. We then recall the definition of weak alternating automata on ranked trees. These will be used to specify properties of Böhm trees. Finally, we introduce the notion of greatest fixpoint model for λY-calculus, and recall what properties of Böhm trees those models capture.

The *set of types* $\mathcal{T}$ is constructed from a unique *basic type* 0 using a binary operation $\to$ that associates to the right. Thus 0 is a type and if A, B are types, so is $(A \to B)$. The order of a type is defined by: $order(0) = 0$, and $order(A \to B) = max(1 + order(A), order(B))$. We work with *tree signatures* that are finite sets of *typed constants of order* 1. So as to simplify the notation and without loss of generality, the constants we use will all have type $0 \to 0 \to 0$.

The set of *simply typed λY-terms* is built from the constants in the signature, and constants Y^A, Ω^A for every type A. These stand for the *fixpoint combinator* and *undefined term*, respectively. Apart from constants, for each type A there is a countable set of variables $x^A, y^A, \ldots$. Terms are build from these constants and variables using typed application and λ-abstraction. We shall write sequences of λ-abstractions $\lambda x_1 \ldots \lambda x_n . M$ with only one λ: $\lambda x_1 \ldots x_n . M$. We take for granted the operational semantics of the calculus given by β and δ reductions.

We now define the infinite normal forms of λY-terms: *Böhm trees*. It is immediate from the definition that a Böhm tree of a closed term of type 0 over a tree signature is a potentially infinite binary tree.

Definition 1. *A* Böhm tree *of a term M is obtained in the following way.*

- *If $M \to^*_{\beta\delta} \lambda \boldsymbol{x}.N_0 N_1 \ldots N_k$ with N_0 a variable or a constant then $BT(M)$ is a tree having its root labeled by $\lambda \boldsymbol{x}.N_0$ and having $BT(N_1)$, ..., $BT(N_k)$ as subtrees.*
- *Otherwise $BT(M) = \Omega^A$, where A is the type of M.*

A weak alternating automaton accepts trees over a fixed tree signature Σ. Recall that we assume that all constants in Σ have type $0 \to 0 \to 0$. By the remark above, these automata can run on Böhm trees of closed terms of type 0.

Definition 2. *A* weak alternating tree automaton *over the signature Σ is:*

$$\mathcal{A} = \langle Q, \Sigma, q^0 \in Q, \delta : Q \times \Sigma \to \mathcal{P}(\mathcal{P}(Q) \times \mathcal{P}(Q)), \rho : Q \to \mathbb{N} \rangle$$

where Q is a finite set of states, $q^0 \in Q$ is the initial state and ρ is a rank function. *For q in Q, we call $\rho(q)$* its rank. *The automaton is* weak *in the sense that when (S_0, S_1) is in $\delta_2(q, a)$, then for every q' in $S_0 \cup S_1$, $\rho(q') \leq \rho(q)$.*

Automata will work on Σ-labelled binary trees that are partial functions $t : \{1,2\}^* \dot{\to} \Sigma \cup \{\Omega\}$ such that their domain is a binary tree, and $t(u) = \Omega$ if u is a leaf, and $t(u) \in \Sigma$ otherwise.

The acceptance of a tree is defined in terms of games between two players that we call Eve and Adam. A *play* between Eve and Adam from some node v of a tree t and some state $q \in Q$ proceeds as follows. If v is a leaf then Eve wins if the rank of q is even, and Adam wins if the rank is odd. Otherwise Eve chooses a pair of sets of states $(S_1, S_2) \in \delta(q, a)$; where a is the label of v. Then Adam chooses S_i (for $i = 1, 2$) and a state $q' \in S_i$. The play continues from the i-th son of v and state q'. If the play is infinite then the winner is decided by looking at ranks of states appearing on the play. Due to the weakness of $\mathcal{A}$ the rank of states in a play can never increase, so it eventually stabilizes at some value. Eve wins if this value is even. A tree t is *accepted* by $\mathcal{A}$ from a state q^0 is Eve has a winning strategy in the game started from the root of t and from q^0.

Observe that without a loss of generality we can assume that δ is monotone, i.e. if $(S_0, S_1) \in \delta(q, a)$ and $S_0 \subseteq S_0'$ and $S_1 \subseteq S_1'$ and for all, $q' \in S_0' \cup S_1'$ $\rho(q') \leq \rho(q)$ then, $(S_0', S_1') \in \delta(q, a)$. Indeed adding the transitions needed to satisfy the monotonicity condition does not give Eve more winning possibilities.

An automaton defines a language of closed terms of type 0 whose Böhm trees it accepts: $L(\mathcal{A}) = \{M : M \text{ has type 0, and } BT(M) \text{ accepted by } \mathcal{A} \text{ from } q^0\}$. Observe that $L(\mathcal{A})$ is closed under $\beta\delta$-conversion.

We use standard notions and notations for models for λY-calculus, in particular for *valuation/variable assignment* and of *interpretation of a term* (see [17]). We shall write $[\![M]\!]^{\mathcal{S}}_{\nu}$ for the interpretation of a term M in a model $\mathcal{S}$ with the valuation ν. As usual, we will omit subscripts or superscripts in the notation of the semantic function if they are clear from the context.

The simplest models of λY-calculus are based on monotone functions. A *GFP-model* of a signature Σ is a tuple $\mathcal{S} = \langle \{\mathcal{S}_A\}_{A\in\mathcal{T}}, \rho\rangle$ where $\mathcal{S}_0$ is a finite lattice, called the *base set* of the model, and for every type $A \to B \in \mathcal{T}$, $\mathcal{S}_{A\to B}$ is the lattice $\mathrm{mon}[\mathcal{S}_A \mapsto \mathcal{S}_B]$ of monotone functions from $\mathcal{S}_A$ to $\mathcal{S}_B$ ordered coordinate-wise. The valuation function ρ is required to interpret Ω^A as the greatest element of $\mathcal{S}_A$, and Y^A as the greatest fixpoint of functions in $\mathcal{S}_{A\to A}$. Observe that every $\mathcal{S}_A$ is finite, hence all the greatest fixpoints exist without any additional assumptions on the lattice.

A GFP model $\mathcal{S}$ over the base set $\mathcal{S}_0$ *recognizes a language* L of λY-terms if there is a subset $F \subseteq \mathcal{S}_0$ such that $L = \{M \mid [\![M]\!]^{\mathcal{S}} \in F\}$. The following theorem characterizes the recognizing power of GFP models.

Theorem 1 ([4]). *A language L of λY-terms is recognized by a GFP-model iff it is a boolean combination of languages of recognized by weak automata whose all states have rank 0.*

3 Models for weak automata

Our goal is to capture WMSO with models similarly to Theorem 1. The model we construct will depend only on the states of the automaton and the ranks of those states. We fix a finite set of states Q and a ranking function $\rho : Q \to \mathbb{N}$. Let m be the maximal rank, i.e., the maximal value ρ takes on Q. For every $0 \le k \le m$ we let $Q_k = \{q \in Q : \rho(q) = k\}$ and $Q_{\le k} = \{q \in Q : \rho(q) \le k\}$.

We define by induction on $k \le m$ an applicative structure $\mathcal{D}^k = (\mathcal{D}^k_A)_{A\in\mathrm{types}}$ and a logical relation $\mathcal{L}^k$ (for $0 < k \le m$) between $\mathcal{D}^{k-1}$ and $\mathcal{D}^k$.

For $k = 0$, the model $\mathcal{D}^0$ is just the model of monotone functions over the powerset of Q_0:

$$\mathcal{D}^0_0 = \mathcal{P}(Q_0) \quad \text{and} \quad \mathcal{D}^0_{A\to B} = \mathrm{mon}[\mathcal{D}^0_A \mapsto \mathcal{D}^0_B],$$

For $k \ge 0$ we have:

$$\mathcal{D}^k_0 = \mathcal{P}(Q_{\le k}), \qquad \mathcal{L}^k_0 = \{(R,P) \in \mathcal{D}^{k-1}_0 \times \mathcal{D}^k_0 \ : \ R = P \cap Q_{\le(k-1)}\},$$
$$\mathcal{L}^k_{A\to B} = \{(f_1,f_2) \in \mathcal{D}^{k-1}_{A\to B} \times \mathrm{mon}[\mathcal{D}^k_A \mapsto \mathcal{D}^k_B] \ : \ \forall (g_1,g_2) \in \mathcal{L}^k_A.\ (f_1(g_1), f_2(g_2)) \in \mathcal{L}^k_B\}$$
$$\mathcal{D}^k_{A\to B} = \{f_2 \ : \ \exists f_1 \in \mathcal{D}^{k-1}_{A\to B}.\ (f_1,f_2) \in \mathcal{L}^k_{A\to B})\}$$

Observe that the structures $\mathcal{D}^k$ are defined by a double induction: the outermost on k and the auxiliary induction on the size of the type. Since $\mathcal{L}^k$ is a logical relation between $\mathcal{D}^{k-1}$ and $\mathcal{D}^k$, each $\mathcal{D}^k$ is an applicative structure. Each $\mathcal{D}^k_A$ is ordered. Indeed, each $\mathcal{D}^k$ can be seen as a part of the model of monotone functions over $\mathcal{P}(Q_{\leq k})$. The order in the later is the inclusion in type 0, and the coordinate-wise ordering for other types.

The next lemma is the main technical lemma summarizing the main properties of the model. These properties are expressed in terms of the logical relation $\mathcal{L}^k_A \subseteq \mathcal{D}^{k-1}_A \times \mathcal{D}^k_A$. For notational convenience we write $\mathcal{L}^k_A(d_1)$ for the set of elements d_2 in the relation with d_1: $\mathcal{L}^k_A(d_1) = \{d_2 : \mathcal{L}^k_A(d_1, d_2)\}$. Figure 1 represents schematically some of the properties stated in the lemma.

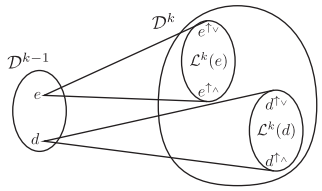

Fig. 1. Relation between models $\mathcal{D}^k$ and $\mathcal{D}^{k+1}$.

Lemma 1. *For every $0 < k \leq m$, and every type A, we have:*

0. *$\mathcal{L}^k_A$ is a lattice: if (d_1, d_2), (e_1, e_2) are in $\mathcal{L}^k_A$, then so are $(d_1 \vee e_1, d_2 \vee e_2)$ and $(d_1 \wedge e_1, d_2 \wedge e_2)$.*
1. *Given (d_1, d_2) and (e_1, e_2) in $\mathcal{L}_{k,A}$, if $d_2 \leq e_2$ then $d_1 \leq e_1$.*
2. *Given d_2 in $\mathcal{D}^k_A$, there is a unique d_1 in $\mathcal{D}^{k-1}_A$, so that (d_1, d_2) in $\mathcal{L}^k_A$. Let us denote this unique element $d_2^{\downarrow}$.*
3. *If $d_1 \leq e_1$ in $\mathcal{D}^{k-1}_A$, then:*
   (a) there is $e_1^{\uparrow\vee}$ in $\mathcal{L}^k_A(e_1)$ so that for every d_2 in $\mathcal{L}^k_A(d_1)$ we have $d_2 \leq e_1^{\uparrow\vee}$,
   (b) there is $d_1^{\uparrow\wedge}$ in $\mathcal{L}^k_A(d_1)$ so that for every e_2 in $\mathcal{L}^k_A(e_1)$ we have $d_2^{\uparrow\wedge} \leq e_2$.

Some consequence of this lemma are spelled out in the following corollaries.

Corollary 1. *For each $k \leq m$, and each type A, $\mathcal{D}^k_A$ is a non-empty lattice.*

The mappings $(\cdot)^{\downarrow}$ and $(\cdot)^{\uparrow\vee}$ form a Galois connection between $\mathcal{D}^k_A$ and $\mathcal{D}^{k-1}_A$. For every $d_2 \in \mathcal{D}^k_A$ and $e_1 \in \mathcal{D}^{k-1}_A$ we have: $d_2^{\downarrow} \leq e_1$ iff $d_2 \leq e_1^{\uparrow\vee}$.

Similarly $(\cdot)^{\downarrow}$ and $(\cdot)^{\uparrow\wedge}$ form an Galois connection between $\mathcal{D}^{k-1}_A$ and $\mathcal{D}^k_A$. For every $d_1 \in \mathcal{D}^{k-1}_A$ and $e_2 \in \mathcal{D}^k_A$ we have: $d_1^{\uparrow\wedge} \leq e_2$ iff $d_1 \leq e_2^{\downarrow}$.

Corollary 2. *Given d_2 in $\mathcal{D}^k_{B\to C}$ and e_2 in $\mathcal{D}^k_B$, we have $(d_2(e_2))^{\downarrow} = d_2^{\downarrow}(e_2^{\downarrow})$. Given d_1 in $\mathcal{D}^{k-1}_{B\to C}$ and e_2 in $\mathcal{D}^k_B$, we have: $d_1^{\uparrow\wedge}(e_2) = (d_1(e_2^{\downarrow}))^{\uparrow\wedge}$ and $d_1^{\uparrow\vee}(e_2) = (d_1(e_2^{\downarrow}))^{\uparrow\vee}$.*

From now, $\bot_A^k$ and $\top_A^k$ will denote the least and the greatest element of $\mathcal{D}_A^k$.

We define an operation $\overline{d}$ on the elements d of $\mathcal{D}_A^k$ by induction on A:

- if $A = 0$, then $\overline{d} = d \cap Q_k$,
- if $A = B \to C$, then for e in $\mathcal{D}^{k,B}$, $\overline{d}(e) = \overline{d(e)}$.

This operation allows interesting decompositions of the elements of $\mathcal{D}_A^k$.

Lemma 2. *For every $0 < k \leq m$, and every d in $\mathcal{D}_A^k$: $d = ((d^{\downarrow})^{\uparrow\wedge}) \vee \overline{d}$ and, letting $\widetilde{d} = (\top_A^{k-1})^{\uparrow\wedge} \vee \overline{d}$, $d = ((d^{\downarrow})^{\uparrow\vee}) \wedge \widetilde{d}$.*

This lemma shows that every element d_2 in $\mathcal{L}_A^k(d_1)$ is of the form $d_1^{\uparrow\wedge} \vee e_2$ with e_2 in $\mathcal{L}_A^k(\bot_A^{k-1})$ and of the form $d_1^{\uparrow\vee} \wedge e_2$ with e_2 in $\mathcal{L}_A^k(\top_A^{k-1})$. Thus not only $\mathcal{L}_A^k$ puts every element d_1 in relation with a lattice $\mathcal{L}^k(d_1)$, but also this lattice is isomorphic to $\mathcal{L}_A^k(\bot_A^{k-1})$ and to $\mathcal{L}_A^k(\top_A^{k-1})$. Therefore, we can see $\mathcal{D}_A^k$ as isomorphic to the lattice $\mathcal{D}_A^{k-1} \times \mathcal{L}_{k,A}(\bot_A^{k-1})$ or to the lattice $\mathcal{D}_A^{k-1} \times \mathcal{L}_{k,A}(\top_A^{k-1})$. This structure is important for type systems in Section 4.

We can now define fixpoint operators in every applicative structure $\mathcal{D}^k$.

Definition 3. *For $f \in \mathcal{D}_{A\to A}^0$ we define*

$$\mathrm{fix}_A^0(f) = \bigwedge\{f^n(\top^0) : n \geq 0\}$$

For $0 < 2k \leq m$ and $f \in \mathcal{D}_{A\to A}^{2k}$ we define

$$\mathrm{fix}_A^{2k}(f) = \bigwedge\{f^n(e) : n \geq 0\} \quad \textit{where} \quad e = (\mathrm{fix}_A^{2k-1}(f^{\downarrow}))^{\uparrow\vee} \ .$$

For $0 < 2k+1 \leq m$ and $f \in \mathcal{D}_{A\to A}^{2k+1}$ we define

$$\mathrm{fix}_A^{2k+1}(f) = \bigvee\{f^n(d) : n \geq 0\} \quad \textit{where} \quad d = (\mathrm{fix}_A^{2k}(f^{\downarrow}))^{\uparrow\wedge} \ .$$

Observe that, for even k, e is obtained with $(\cdot)^{\uparrow\vee}$; while for odd k, $(\cdot)^{\uparrow\wedge}$ is used.

Standard techniques show that equipped with this interpretation of the fixpoint, each $\mathcal{D}^k$ forms a model of the λY-calculus.

Theorem 2. *For every finite set Q and a function $\rho : Q \to \mathbb{N}$. For every $k \geq 0$ the applicative structure $\mathcal{D}^k$ is a model of the λY-calculus.*

For a λY-term M, and a valuation ν, we write $[\![M]\!]_\nu^k$ for the interpretation of M in $\mathcal{D}^k$ with the valuation ν.

Now fix a weak alternating automaton $\mathcal{A} = \langle Q, \Sigma, \delta, \rho\rangle$ as defined in Section 2. We let the semantics of constants be

$$[\![a]\!]^k(S_0, S_1) = \{q : (S_0, S_1) \in \delta(q, a)\} \ .$$

Thanks to our assumption about monotonicity of δ, such a function is monotone.

For a term M consider the set of states from which $\mathcal{A}$ accepts the tree $BT(M)$.

$$\mathcal{A}(M) = \{q \in Q : \mathcal{A} \text{ accepts } BT(M) \text{ from } q\}$$

We show that our model $\mathcal{D}^m$ can calculate $\mathcal{A}(M)$; here m is the maximal value of the rank function of $\mathcal{A}$. For the proof we adapt approximation and adequacy techniques that are standard in λY-calculus. In particular, adequacy exploits the stratification of the model in an original manner. It allows to work solely with finitary objects and invariants expressed by a logical relation.

Theorem 3. *For every closed term M of type 0, for every $0 \leq k \leq m$ we have $[\![M]\!]^k = \mathcal{A}(M) \cap Q_{\leq k}$.*

4 From models to type systems

Throughout this section we assume that we work with a fixed signature Σ and a weak alternating automaton $\mathcal{A} = \langle Q, \Sigma, \delta, \rho \rangle$. Moreover, in the context of type systems it will be more convenient to consider Y as a variable binder and not as a term. The semantics of $Yx.M$ is the same as that of $Y(\lambda x.M)$.

We define a type system to reason about the values of λ-terms in the model. We follow Abramsky's idea of domains in logical form [2] further adapted to the typed setting in [18]. For this we first give a syntactic representation of elements of the model akin to intersection types. The sets types^k_A and Types^k_A are defined by induction the structure of A:

$$\begin{aligned} \mathrm{types}^0_k &= \{q \in Q : \rho(q) = k\} \ , \\ \mathrm{types}^k_{A\to B} &= \{T \to s : T \subseteq \mathrm{Types}^k_A \text{ and } s \in \mathrm{types}^k_B\} \ , \\ \mathrm{Types}^k_A &= \bigcup_{0\leq l\leq k} \mathrm{types}^l_A \ . \end{aligned}$$

When we write types_A or Types_A we mean types^m_A and Types^m_A respectively; where m is the maximal rank used by the automaton $\mathcal{A}$. Moreover, for $S \subseteq \mathrm{Types}^k_A$ and $T \subseteq \mathrm{Types}^k_B$ we write $S \to T$ for $\{S \to t : t \in T\}$. Notice that $S \to T$ is included in $\mathrm{Types}^k_{A\to B}$.

Before going further, we need to recall the notions of step and co-step functions that are particular monotone functions from a lattice $\mathcal{L}_1$ to a lattice $\mathcal{L}_2$. For d in $\mathcal{L}_1$ and e in $\mathcal{L}_2$, the *step function* $d \rightharpoonup e$ and the *co-step function* $d \rightharpoondown g$ are defined by:

$$(d \rightharpoonup e)(h) = \begin{cases} e \text{ when } d \leq h \\ \bot \text{ otherwise} \end{cases} \qquad (d \rightharpoondown e)(h) = \begin{cases} e \text{ when } h \leq d \\ \top \text{ otherwise} \end{cases}$$

To emphasize that we work in $\mathcal{D}^l$ we will write $d \rightharpoonup^l e$ and $d \rightharpoondown^l e$.

Types can be meaningfully interpreted at every level of the model. So $[\![t]\!]^l$ will denote the interpretation of t in $\mathcal{D}^l$ defined as follows.

$$[\![q]\!]^l = \begin{cases} \{q\} & \text{if the rank of } q \text{ is at most } l \\ \emptyset & \text{otherwise} \end{cases}$$

$$[\![S]\!]^l = \bigvee \{[\![t]\!]^l : t \in S\} \qquad \text{for } S \subseteq \text{Types}_A$$

$$[\![T \to s]\!]^l = [\![T]\!]^l \rightarrow^l [\![s]\!]^l \qquad \text{for } (T \to s) \in \text{Types}_A$$

We have $[\![S_1 \cup S_2]\!]^l = [\![S_1]\!]^l \vee [\![S_2]\!]^l$ and moreover $[\![S \to T]\!]^l = [\![S]\!]^l \rightarrow^l [\![T]\!]^l$.

Lemma 3. *For every type A, if $S \subseteq \text{Types}_A$ and $k \leq m$ we have: $[\![S]\!]^k = [\![S \cap \text{Types}_A^k]\!]^k$, $[\![S \cap \text{Types}_A^k]\!]^{k+1} = ([\![S]\!]_A^k)^{\uparrow\wedge}$ and $[\![S]\!]^k = ([\![S]\!]^{k+1})^{\downarrow}$.*

We can now show that every element of $\mathcal{D}_A^k$ is representable by a type. For this we use $\overline{(\cdot)}$ operation defined on page 7.

Lemma 4. *For every $k \leq m$ and every type A. For every d in $\mathcal{D}_A^k$ there is $S \subseteq \text{Types}_A^k$ so that $[\![S]\!]_k = d$, and there is $S' \subseteq \text{types}_A^k$ so that $[\![S']\!]_k = \overline{d}$.*

We now give subsumption rules that simulate on types the ordering in $\mathcal{D}_A^k$. So as to make the connection with the order in the model clearer, we have adopted an ordering of intersection types that is dual to the usual one.

$$\frac{S \subseteq T \subseteq Q}{S \sqsubseteq_0 T} \qquad \frac{\forall s \in S, \exists t \in T, s \sqsubseteq_A t}{S \sqsubseteq_A T} \qquad \frac{T \sqsubseteq_A S \quad s \sqsubseteq_B t}{S \to s \sqsubseteq_{A \to B} T \to t}$$

Lemma 5. *For every k, A, and $S, T \subseteq \text{Types}_A^k$: $[\![S]\!]^k \leq [\![T]\!]^k$ iff $S \sqsubseteq_A T$.*

The typing system presented in Figure 2 derives judgments of the form $\Gamma \vdash M \geq S$ where Γ is an environment containing all free variables of the term M and $S \subseteq \text{Types}_A$ with A the type of M. As usual, an environment Γ is a finite list $x_1 \geq S_1, \ldots, x_n \geq S_n$ where x_i is a variable of type A_i, and $S_i \subseteq \text{Types}_{A_i}$. We will use a functional notation and write $\Gamma(x_i)$ for S_i. We shall also write $\Gamma, x \geq S$ with its usual meaning. Given S included in $\text{Types}_{A \to B}$ and T included in Types_A we write $S(T)$ for the set $\{t : (U \to t) \in S \wedge U \sqsubseteq T\}$.

Lemma 6. *For $S \subseteq \text{Types}_{A \to B}^k$ and $T \subseteq \text{Types}_A^k$, we have: $[\![S(T)]\!]^k = [\![S]\!]^k([\![T]\!]^k)$.*

We now present the second main result of the paper saying that the type system can derive all lower-approximations of meanings of terms in the model. For an environment Γ, we write $[\![\Gamma]\!]^k$ for the valuation so that $[\![\Gamma]\!]^k(x) = [\![\Gamma(x)]\!]^k$.

Theorem 4. *For $S \subseteq \text{Types}^k$: $[\![M]\!]^k_{[\![\Gamma]\!]^k} \geq [\![S]\!]^k$ iff $\Gamma \vdash M \geq S$ is derivable.*

A consequence of this theorem and of Theorem 3 is that for closed terms M of type 0: $\Gamma \vdash M \geq \{q\}$ is derivable iff $BT(M)$ is accepted by $\mathcal{A}$ from state q.

$$\frac{}{\Gamma, x \geq S \vdash x \geq S} \qquad \frac{\Gamma \vdash M \geq S \quad \Gamma \vdash M \geq T}{\Gamma \vdash M \geq S \cup T} \qquad \frac{\Gamma \vdash M \geq S \quad T \sqsubseteq S}{\Gamma \vdash M \geq T}$$

$$\frac{(S_1, S_2) \in \delta(a, q)}{\Gamma \vdash a \geq \{S_1 \to S_2 \to q\}}$$

$$\frac{\Gamma \vdash M \geq S \quad \Gamma \vdash N \geq T}{\Gamma \vdash MN \geq S(T)} \qquad \frac{S \subseteq \mathrm{Types}^k,\, T \subseteq \mathrm{types}^k \quad \Gamma, x \geq S \vdash M \geq T}{\Gamma \vdash \lambda x.M \geq S \to T}$$

$$\frac{S, T \subseteq \mathrm{Types}_A^{2k+1}, \quad \Gamma \vdash (\lambda x.M) \geq S \quad \Gamma \vdash (Yx.M) \geq T}{\Gamma \vdash Yx.M \geq S(T)}\, Y \text{ odd}$$

$$\frac{S \subseteq \mathrm{types}_A^{2k}, \quad T \subseteq \mathrm{Types}_A^{2k-1}, \quad \Gamma \vdash \lambda x.M \geq (S \cup T) \to S \quad \Gamma \vdash Yx.M \geq T}{\Gamma \vdash Yx.M \geq S \cup T}\, Y \text{ even}$$

Fig. 2. Type system

As we have seen, the applicative structure $\mathcal{D}_A^k$ is a lattice, therefore each construction can be dualized: in Abramsky's methodology, this consists in considering $\wedge$-prime elements of the models, meets and co-step functions instead of $\vee$-primes, joins and step functions. It is worse noticing that dualizing at the level of the model amounts to dualizing the automaton. So in particular, we can define a system so that $BT(M)$ is not accepted by $\mathcal{A}$ from state q iff $\Gamma \vdash M \not\geq q$ is derivable. While the first typing system establishes positive facts about the semantics, the second one refutes them. For this, we use the same syntax to denote types, but we give types a different semantics that is dual to the first semantics we have used.

$$\langle\!\langle q \rangle\!\rangle^k = Q_{\leq k} - \{q\}\ , \qquad \langle\!\langle S \to f \rangle\!\rangle^k = \Big(\bigwedge \{\langle\!\langle g \rangle\!\rangle^k : g \in S\}\Big) \rightharpoonup^k \langle\!\langle f \rangle\!\rangle^k\ .$$

The dual type system is presented in Figure 3. The notation is as before but we use $\not\geq$ instead of $\geq$. Similarly to the definition of $[\![\cdot]\!]^k$, we write $\langle\!\langle S \rangle\!\rangle^k$ for $\bigwedge\{\langle\!\langle s \rangle\!\rangle^k : s \in S\}$ and we also have that $\langle\!\langle T \to S \rangle\!\rangle^k = \langle\!\langle T \rangle\!\rangle^k \rightharpoonup^k \langle\!\langle S \rangle\!\rangle^k$. We also define $S(T)$ to be $\{s : U \to s \in S \wedge U \sqsupseteq T\}$. With those notations, the rules for application, abstraction and variable do not change and are not presented in the Figure 2. By duality from Theorem 4 we obtain:

Theorem 5. *For $S \subseteq \mathrm{Types}^k$: $\Gamma \vdash M \not\geq S$ is derivable iff $[\![M]\!]^k_{[\![\Gamma]\!]^k} \leq \langle\!\langle S \rangle\!\rangle^k$.*

As an interesting corollary of Theorems 4 and 5 we have:

Corollary 3. *For a closed term M of type 0:*

$$[\![M]\!] = [\![S]\!] \quad \textit{iff} \quad \textit{both} \vdash M \geq S \textit{ and} \vdash M \not\geq (Q - S).$$

$$\frac{S \subseteq T \subseteq Q}{S \sqsupseteq_0 T} \qquad \frac{\forall s \in S, \exists t \in T, s \sqsupseteq_A t}{S \sqsupseteq_A T} \qquad \frac{T \sqsupseteq_A S \quad s \sqsupseteq_B t}{S \to s \sqsupseteq_{A\to B} T \to t}$$

$$\frac{\forall (S_1, S_2) \in \delta(a, q), (T_1 \cap S_1) \cup (T_2 \cap S_2) \neq \emptyset}{\Gamma \vdash a \not\geq T_1 \to T_2 \to q}$$

$$\frac{S \subseteq \mathrm{types}_A^{2k+1}, \quad T \in \mathrm{Types}_A^{2k}, \quad \Gamma \vdash \lambda x.M \not\geq (S \cup T) \to S \quad \Gamma \vdash Yx.M \not\geq T}{\Gamma \vdash Yx.M \not\geq S \cup T} \; Y \text{ odd}$$

$$\frac{S, T \in \mathrm{Types}_A^{2k}, \quad \Gamma \vdash (\lambda x.M) \not\geq S \quad \Gamma \vdash (Yx.M) \not\geq T}{\Gamma \vdash Yx.M \not\geq S(T)} \; Y \text{ even}$$

Fig. 3. Dual type system

5 Conclusion

We have shown how to construct a model for a given weak alternating tree automaton so that the value of a term in the model determines if the Böhm tree of the term is accepted by the automaton. Our construction builds on ideas from [4] but requires a more modular construction of the model. The model turns out to have very rich structure, as testified by Galois connections (Corollary 1) and its decomposition principles (Lemma 2). This structure allows us to derive simple type systems for WMSO properties following the *domains in logical form* approach.

Our model construction allows to immediately deduce reflection [3] and transfer [6] theorems for WMSO. This gives a unifying and, arguably, very simple proofs of these theorems.

The idea behind the reflection construction is to transform a given term so that at every moment of its evaluation every subterm "knows" its meaning in the model. In [3] this property is formulated slightly differently and is proved using a detour to higher-order pushdown automata. Recently Haddad [5] has given a direct proof for all MSO properties. The proof is based on some notion of applicative structure that is less constrained than a model of the λY-calculus. One could apply his construction, or take the one from [4] where we have constructed models for divergence checking. We have given there a construction of a reflective term that works for all finitary models of the λY-calculus, in particular for those presented in this paper.

The transfer theorem says that for a fixed finite vocabulary of terms, an MSOL formula φ can be effectively transformed into an MSOL formula $\widehat{\varphi}$ such that for every term M of type 0 over the fixed vocabulary: M satisfies $\widehat{\varphi}$ iff the Böhm tree of M satisfies φ. Since the MSO theory of a term, that is a finite graph, is decidable, the transfer theorem implies decidability of MSO theory of Böhm trees of λY-terms. As we have shown in [6] it gives also a number of other results.

A transfer theorem for WMSO can be deduced from our model construction. For every WMSO formula φ we need to find a formula $\widehat{\varphi}$ as above. For this we transform φ into a weak alternating automaton $\mathcal{A}$, and construct a model $\mathcal{D}_\varphi$ based on $\mathcal{A}$. Thanks to the restriction on the vocabulary, it is quite easy to write for every element d of the model $\mathcal{D}_\varphi$ a WMSO formula α_d such that for every term of type 0 in the restricted vocabulary: $M \vDash \alpha_d$ iff $[\![M]\!]^{\mathcal{D}} = d$. The formula $\widehat{\varphi}$ is then just a disjunction $\bigvee_{d \in F} \alpha_d$, where F is the set elements of $\mathcal{D}$ characterizing terms whose Böhm tree satisfies φ.

The fixpoints in our model are non-extremal: they are neither the least nor the greatest fixpoints. From [4] we know that this is unavoidable. We are aware of very few works considering such cases. Our models are an instance of cartesian closed categories with internal fixpoint operation as studied by Bloom and Esik [19]. Our model satisfies not only Conway identities but also a generalization of the *commutative axioms* of iteration theories [20]. Thus it is possible to give semantics to the infinitary λ-calculus in our models. It is an essential step towards obtaining an algebraic framework for weak regular languages [21].

References

1. Ong, C.H.L.: On model-checking trees generated by higher-order recursion schemes. In: LICS. (2006) 81–90
2. Abramsky, S.: Domain theory in logical form. Ann. Pure Appl. Logic **51**(1-2) (1991) 1–77
3. Broadbent, C., Carayol, A., Ong, L., Serre, O.: Recursion schemes and logical reflection. In: LICS. (2010) 120–129
4. Salvati, S., Walukiewicz, I.: Using models to model-check recursive schemes. In: TLCA. Volume 7941 of LNCS., Springer (2013) 189–204
5. Haddad, A.: Model checking and functional program transformations. In: FSTTCS. Volume 24 of LIPIcs., Schloss Dagstuhl (2013) 115–126
6. Salvati, S., Walukiewicz, I.: Evaluation is MSOL-compatible. In: FSTTCS. Volume 24 of LIPIcs., Schloss Dagstuhl (2013) 103–114
7. Terui, K.: Semantic evaluation, intersection types and complexity of simply typed lambda calculus. In: RTA. Volume 15 of LIPIcs., Schloss Dagstuhl (2012) 323–338
8. Muller, D.E., Schupp, P.E.: The theory of ends, pushdown automata, and second-order logic. Theor. Comput. Sci. **37** (1985) 51–75
9. Knapik, T., Niwinski, D., Urzyczyn, P.: Higher-order pushdown trees are easy. In: FoSSaCS. Volume 2303. (2002) 205–222
10. Hague, M., Murawski, A.S., Ong, C.H.L., Serre, O.: Collapsible pushdown automata and recursion schemes. In: LICS, IEEE Computer Society (2008) 452–461
11. Kobayashi, N., Ong, L.: A type system equivalent to modal mu-calculus model checking of recursion schemes. In: LICS. (2009) 179–188
12. Salvati, S., Walukiewicz, I.: Krivine machines and higher-order schemes. In: ICALP (2). Volume 6756 of LNCS., Springer (2011) 162–173
13. Aehlig, K.: A finite semantics of simply-typed lambda terms for infinite runs of automata. Logical Methods in Computer Science **3**(1) (2007) 1–23
14. Kobayashi, N.: Model checking higher-order programs. J. ACM **60**(3) (2013) 20–89
15. Amadio, R.M., Curien, P.L.: Domains and Lambda-Calculi. Volume 46 of Cambridge Tracts in Theoretical Computer Science. Cambridge University Press (1998)

16. Ramsay, S.J., Neatherway, R.P., Ong, C.H.L.: A type-directed abstraction refinement approach to higher-order model checking. In: POPL, ACM (2014) 61–72
17. Hindley, J.R., Seldin, J.P.: Lambda-Calculus and Combinators. Cambridge University Press (2008)
18. Salvati, S., Manzonetto, G., Gehrke, M., Barendregt, H.: Loader and Urzyczyn are logically related. In: ICALP (2). Volume 7392 of LNCS., Springer (2012) 364–376
19. Bloom, S.L., Ésik, Z.: Fixed-point operations on CCC's. part I. Theoretical Computer Science **155** (1996) 1–38
20. Bloom, S.L., Ésik, Z.: Iteration Theories: The Equational Logic of Iterative Processes. EATCS Monographs in Theoretical Computer Science. Springer (1993)
21. Blumensath, A.: An algebraic proof of Rabin's tree theorem. Theor. Comput. Sci. **478** (2013) 1–21

A Model for weak automata

In this part of the appendix we show the properties of the model. We start by showing that the logical relation $\mathcal{L}^k$ has a lattice structure at each type, i.e. $\mathcal{L}^k_A$ is a lattice. We split the proof of Lemma 1 into the proof of two different Lemmas.

Lemma 7. *For every $0 < k \leq m$ and every type A, $\mathcal{L}^k_A$ is a lattice: if (d_1, d_2), (e_1, e_2) are in $\mathcal{L}^k_A$, then so are $(d_1 \vee e_1, d_2 \vee e_2)$ and $(d_1 \wedge e_1, d_2 \wedge e_2)$.*

Proof. For a fixed k, we show this property by induction on A. The case $A = 0$ is obvious. In case $A = B \to C$, let us take (f_1, f_2) in $\mathcal{L}^k_B$. By definition of logical relations, we have that $(d_1(f_1), d_2(f_2))$ and $(e_1(f_1), e_2(f_2))$ are in $\mathcal{L}^k_C$. Therefore, by induction hypothesis, $(d_1(f_1) \vee e_1(f_1), d_2(f_2) \vee e_2(f_2)) = ((d_1 \vee d_2)(f_1), (d_2 \vee e_2)(f_2))$ and $((d_1 \wedge e_1)(f_1), (d_2 \wedge e_2)(f_2))$ are in $\mathcal{L}^k_C$. As this holds for every (f_1, f_2) in $\mathcal{L}^k_B$, this shows that $(d_1 \vee e_1, d_2 \vee e_2)$ and $(d_1 \wedge e_1, d_2 \wedge e_2)$ are indeed in $\mathcal{L}^k_A$. □

We now prove the lemma that states that $\mathcal{L}^k_A(d)$ is a non-empty lattice and that induces the Galois connections between $\mathcal{D}^{k-1}$ and $\mathcal{D}^k$ that are explained in Corollary 1.

Lemma 8. *For every $0 < k \leq m$, and every type A, we have:*

1. *Given (d_1, d_2) and (e_1, e_2) in $\mathcal{L}_{k,A}$, if $d_2 \leq e_2$ then $d_1 \leq e_1$.*
2. *Given d_2 in $\mathcal{D}^k_A$, there is a unique d_1 in $\mathcal{D}^{k-1}_A$, so that (d_1, d_2) in $\mathcal{L}^k_A$. Let us denote this unique element $d_2^{\downarrow}$.*
3. *If $d_1 \leq e_1$ in $\mathcal{D}^{k-1}_A$, then:*
   (a) there is $e_1^{\uparrow\vee}$ in $\mathcal{L}^k_A(e_1)$ so that for every d_2 in $\mathcal{L}^k_A(d_1)$ we have $d_2 \leq e_1^{\uparrow\vee}$,
   (b) there is $d_1^{\uparrow\wedge}$ in $\mathcal{L}^k_A(d_1)$ so that for every e_2 in $\mathcal{L}^k_A(e_1)$ we have $d_2^{\uparrow\wedge} \leq e_2$.

Proof. The proof is by simultaneous induction on the size of A.

Notice that the second item is an immediate consequence of the first one. We shall therefore not prove it, but we feel free to use it as an induction hypothesis.

We start with the case when A is the base type 0:
Ad 1. In that case $\mathcal{D}^k_A = Q_{\leq k}$, $e_1 = e_2 \cap Q_{\leq k-1}$ and $d_1 = d_2 \cap Q_{\leq k-1}$. Thus, we indeed have that $d_2 \leq e_2$ implies that $d_1 \leq e_1$.
Ad 3. Here, we have $\mathcal{D}^{k-1}_A = Q_{\leq k-1}$ and then letting $e_1^{\uparrow\vee} = e_1 \cup Q_k$ and $d_2^{\uparrow\wedge} = d_2$ is enough to conclude.

Let us now suppose that $A = B \to C$:
Ad 1. Given f_1 in $\mathcal{D}^{k-1}_B$, by induction hypothesis, using item 3, we know that there exists f_2 in $\mathcal{D}^k_B$ so that (f_1, f_2) is in $\mathcal{L}^k_B$. Thus, we have $(d_1(f_1), d_2(f_2))$ and $(e_1(f_1), e_2(f_2))$ in $\mathcal{L}^k_C$. With the assumption that $d_2 \leq e_2$, we obtain $d_2(f_2) \leq e_2(f_2)$. By induction hypothesis we get $e_1(f_1) \leq d_1(f_1)$. As f_1 is arbitrary, we can conclude that $e_1 \leq d_1$.

Ad 3. By induction hypothesis, using item 2, for f_2 in $\mathcal{D}^k_B$, there is a unique element $f_2^{\downarrow}$ of $\mathcal{D}^{k-1}_B$ so that $(f_2^{\downarrow}, f_2)$ is in $\mathcal{L}^k_B$. Given h_1 in $\mathcal{D}^{k-1}_A$ we define for every element f_2 in $\mathcal{D}^k_B$:

$$h_1^{\uparrow\vee}(f_2) = (h_1(f_2^{\downarrow}))^{\uparrow\vee} \qquad h_1^{\uparrow\wedge}(f_2) = (h_1(f_2^{\downarrow}))^{\uparrow\wedge} \ .$$

We will verify only item 3(a), item 3(b) being analogous.

We need to check that $h_1^{\uparrow\vee}$ is in $\mathcal{D}^k_A$. First of all we need to check that it is in $\mathrm{mon}[\mathcal{D}^k_B \mapsto \mathcal{D}^k_C]$. Take g_2 and f_2 in $\mathcal{D}^k_B$ so that $g_2 \leq f_2$. By induction hypothesis, using item 1, we have that $g_2^{\downarrow} \leq f_2^{\downarrow}$. Then $h_1(g_2^{\downarrow}) \leq h_1(f_2^{\downarrow})$ by monotonicity of h_1. From item 3 of induction hypothesis we obtain $(h_1(g_2^{\downarrow}))^{\uparrow\vee} \leq (h_1(f_2^{\downarrow}))^{\uparrow\vee}$; proving that $h_1^{\uparrow\vee}$ is monotone.

Next, we show that $(h_1, h_1^{\uparrow\vee})$ is in $\mathcal{L}^k_A$. If we take $(f_2^{\downarrow}, f_2)$ in $\mathcal{L}^k_B$, we have, by induction hypothesis, item 3, that $(h_1(f_2^{\downarrow}), (h_1(f_2^{\downarrow}))^{\uparrow\vee})$ is in $\mathcal{L}^k_C$. But, by our definition, this implies that $(h_1(f_2^{\downarrow}), h_1^{\uparrow\vee}(f_2))$ is in $\mathcal{L}^k_C$. As f_2 is arbitrary we obtain that $(h_1, h_1^{\uparrow\vee})$ is indeed in $\mathcal{L}^k_A$ proving then that $h_1^{\uparrow\vee}$ is in $\mathcal{D}^k_A$.

It remains to prove that, given $d_1 \leq e_1$ in $\mathcal{D}^k_A$, for all d_2 in $\mathcal{L}^k_A(d_1)$ we have $d_2 \leq e_1^{\uparrow\vee}$. Given $(f_2^{\downarrow}, f_2)$ in $\mathcal{L}^k_B$, we have $(d_1(f_2^{\downarrow}), d_2(f_2))$ in $\mathcal{L}^k_C$. Using the induction hypothesis, item 3, we get $d_2(f_2) \leq (d_1(f_2^{\downarrow}))^{\uparrow\vee}$. Since $d_1 \leq e_1$ we obtain $d_2(f_2) \leq (e_1(f_2^{\downarrow}))^{\uparrow\vee}$ that is the desired $d_2(f_2) \leq e_1^{\uparrow\vee}(f_2)$. As f_2 is arbitrary, this shows that $d_2 \leq e_1^{\uparrow\vee}$. □

We now turn to prove how elements of $\mathcal{D}^k_A$ can be decomposed as pairs (d, e) where d is in $\mathcal{D}^{k-1}_A$ and in $\mathcal{L}^k_A(\bot^k_A)$ or $\mathcal{L}^k_A(\top^k_A)$. This decomposition plays a central role in the definition of the type systems given in Section 4.

Lemma 2 *For every $0 < k \leq m$, and every d in $\mathcal{D}^k_A$: $d = ((d^{\downarrow})^{\uparrow\wedge}) \vee \overline{d}$ and, letting $\widetilde{d} = (\top^{k-1}_A)^{\uparrow\wedge} \vee \overline{d}$, $d = ((d^{\downarrow})^{\uparrow\vee}) \wedge \widetilde{d}$.*

Proof. We prove the first identity by induction on A.

In case $A = 0$, from definitions we get $\overline{d} = d \cap Q_k$ while $(d^{\downarrow})^{\uparrow\wedge} = d \cap Q_{\leq k-1}$. Thus we indeed have $d = ((d^{\downarrow})^{\uparrow\wedge}) \vee \overline{d}$.

In case $A = B \to C$. Given e in $\mathcal{D}^k_B$, we have that $(d^{\downarrow})^{\uparrow\wedge}(e) = (d^{\downarrow}(e^{\downarrow}))^{\uparrow\wedge} = \underline{((d(e))^{\downarrow})^{\uparrow\wedge}}$. Moreover, $\overline{d}(e) = \overline{d(e)}$. Therefore, $(((d^{\downarrow})^{\uparrow\wedge}) \vee \overline{d})(e) = (((d(e))^{\downarrow})^{\uparrow\wedge}) \vee \overline{d(e)}$. But, by induction, we have $d(e) = (((d^{\downarrow})^{\uparrow\wedge}) \vee \overline{d})(e) = (((d(e))^{\downarrow})^{\uparrow\wedge}) \vee \overline{d(e)}$. As e is arbitrary, we get the identity.

The second identity is also obtained by induction on A.

In case $A = 0$, $\widetilde{d} = Q_{\leq k-1} \cup (d \cap Q_k)$ and $(d^{\downarrow})^{\uparrow\vee} = Q_k \cup (d \cap Q_{\leq k-1})$ and therefore $(d^{\downarrow})^{\uparrow\vee} \wedge \widetilde{d} = d$.

In case $A = B \to C$, given e in $\mathcal{D}^k_B$, we have that $(d^{\downarrow})^{\uparrow\vee}(e) = (d^{\downarrow}(e^{\downarrow}))^{\uparrow\vee} = (d(e)^{\downarrow})^{\uparrow\vee}$. Moreover, $(\top^{k-1}_A)^{\uparrow\wedge}(e) = (\top^{k-1}_A(e)^{\downarrow})^{\uparrow\wedge} = (\top^{k-1}_B)^{\uparrow\wedge}$ and $\overline{d}(e) = \overline{d(e)}$. Thus, $\widetilde{d}(e) = (\top^{k-1}_B)^{\uparrow\wedge} \vee \overline{d(e)} = \widetilde{d(e)}$ and thus $((d^{\downarrow})^{\uparrow\vee} \wedge \widetilde{d})(e) = (d(e)^{\downarrow})^{\uparrow\vee} \wedge \widetilde{d(e)}$ which is equal, by induction to $d(e)$. As e is arbitrary, we obtain that the identity holds. □

We now turn to show that equipped with the interpretation of fixpoints given by Definition 3 the applicative structure $\mathcal{D}^k$ is a model of the λY-calculus. First, we check that fix^k_A is indeed an element of the model and that it is a fixpoint.

Lemma 9. *For every $0 \leq k \leq m$ and every type A we have that fix^k_A is monotone, and if $k > 0$ then $(\mathrm{fix}^{k-1}_A, \mathrm{fix}^k_A) \in \mathcal{L}^k_{(A\to A)\to A}$. Moreover for every f in $\mathcal{D}^k_{A\to A}$, $f(\mathrm{fix}^k_A(f)) = \mathrm{fix}^k_A(f)$.*

Proof. For 0 the statement is obvious. We will only consider the case where k is even, the other being dual.

Consider the case $2k > 0$. First we show monotonicity. Suppose $g \leq h$ are two elements of $\mathcal{D}^{2k}_{A\to A}$. By Lemma 1 we get $g^\downarrow \leq h^\downarrow$. Consider $g^1 = \mathrm{fix}^{2k-1}_A(g^\downarrow)$ and $h^1 = \mathrm{fix}^{2k-1}_A(h^\downarrow)$. By induction hypothesis fix^{2k-1}_A is monotone, so $g^1 \leq h^1$. Then, once again using the Lemma 1, we have $(g^1)^{\uparrow\vee} \leq (h^1)^{\uparrow\vee}$. This implies $\mathrm{fix}^{2k}_A(g) \leq \mathrm{fix}^{2k}_A(h)$.

Now we show $(\mathrm{fix}^{2k-1}_A, \mathrm{fix}^{2k}_A) \in \mathcal{L}^{2k}_{(A\to A)\to A}$. We take arbitrary $(f_1, f_2) \in \mathcal{L}^{2k}_{A\to A}$ and we need to show $(\mathrm{fix}^{2k-1}_A(f_1), \mathrm{fix}^{2k}_A(f_2)) \in \mathcal{L}^{2k}_A$. This follows from the following calculation

$$\begin{aligned}
(\mathrm{fix}^{2k-1}(f_1), (\mathrm{fix}^{2k-1}(f_1))^{\uparrow\vee}) &\in \mathcal{L}^{2k}_A && \text{by Lemma 1}\\
(f_1(\mathrm{fix}^{2k-1}(f_1)), f_2((\mathrm{fix}^{2k-1}(f_1)^{\uparrow\vee})) &\in \mathcal{L}^{2k}_A && \text{by logical relation}\\
(\mathrm{fix}^{2k-1}(f_1), f_2((\mathrm{fix}^{2k-1}(f_1)^{\uparrow\vee})) &\in \mathcal{L}^{2k}_A && \text{since } \mathrm{fix}^{2k-1}(f_1) \text{ is a fixpoint of } f_1\\
(\mathrm{fix}^{2k-1}(f_1), f_2^i((\mathrm{fix}^{2k-1}(f_1)^{\uparrow\vee})) &\in \mathcal{L}^{2k}_A && \text{for every } i \geq 0
\end{aligned}$$

Moreover, from Lemma 1, we have that $f_2((\mathrm{fix}^{2k-1}(f_1)^{\uparrow\vee}) \leq (\mathrm{fix}^{2k-1}(f_1))^{\uparrow\vee}$ which then implies that for every $i \in \mathbb{N}$, $f_2^{i+1}((\mathrm{fix}^{2k-1}(f_1)^{\uparrow\vee}) \leq f_2^i((\mathrm{fix}^{2k-1}(f_1))^{\uparrow\vee})$. Therefore, $(f_2^i((\mathrm{fix}^{2k-1}(f_1))^{\uparrow\vee})_{i\in\mathbb{N}}$ is a decreasing sequence of $\mathcal{D}^{2k}_A$. Since the model is finite this sequence reaches the fixpoint, namely $\mathrm{fix}^{2k}_A(f_2) = f_2^i((\mathrm{fix}^{2k-1}_A(f_1)^{\uparrow\vee}))$ for some i. Thus, at the same time, this shows that $(\mathrm{fix}^{2k-1}_A(f_1), \mathrm{fix}^{2k}_A(f_2)) \in \mathcal{L}^{2k}_A$ and that $\mathrm{fix}^{2k}(f_2)$ is a fixpoint of f_2. □

This lemma has the following interesting corollary that will prove useful in the study of type systems.

Corollary 4. *For $k > 0$, A a type, and $f \in \mathcal{D}^k_{A\to A}$ we have*

$$\begin{aligned}
\mathrm{fix}^{2k}_A(f) &= \bigvee\{d \mid f(d) \geq d \text{ and } d^\downarrow = \mathrm{fix}^{2k-1}_A(f)\}\\
\mathrm{fix}^{2k+1}_A(f) &= \bigwedge\{d \mid f(d) \leq d \text{ and } d^\downarrow = \mathrm{fix}^{2k}_A(f)\}
\end{aligned}$$

Now we turn to showing that for every k, $\mathcal{D}^k$ is indeed a model of λY-calculus. Since $\mathcal{D}^k_{A\to B}$ does not contain all the functions from $\mathcal{D}^k_A$ to $\mathcal{D}^k_B$ we must show that there are enough of them to form a model of λY, the main problem being to show that $[\![\lambda x.M]\!]^{\mathcal{D}^k}_v$ defines an element of $\mathcal{D}^k$. For this it will

be more appropriate to consider the semantics of a term as a function of values of its free variables. Given a finite sequence of variables $\boldsymbol{x} = x_1, \ldots, x_n$ of types $A_1, \ldots, A_n$ respectively and a term M of type B with free variables in $\boldsymbol{x}$, the meaning of M in the model $\mathcal{D}^k$ with respect to $\boldsymbol{x}$ will be a function $[\![M]\!]^k_{\boldsymbol{x}}$ in $\mathcal{D}^k_{A_1} \to \cdots \to \mathcal{D}^k_{A_n} \to \mathcal{D}^k_B$ that represents the function $\lambda \boldsymbol{p}.[\![M]\!]^k_{[p_1/x_1,\ldots,p_n/x_n]}$. Formally it is defined as follows:

1. $[\![Y^B]\!]^k_{\boldsymbol{x}} = \lambda \boldsymbol{p}.\ \mathrm{fix}_B$
2. $[\![a]\!]^k_{\boldsymbol{x}} = \lambda \boldsymbol{p}.\ \rho(a)$
3. $[\![x_i^{A_i}]\!]^k_{\boldsymbol{x}} = \lambda \boldsymbol{p}.p_i$
4. $[\![MN]\!]^k_{\boldsymbol{x}} = \lambda \boldsymbol{p}.\ ([\![M]\!]^k_{\boldsymbol{x}}\ \boldsymbol{p})([\![N]\!]^k_{\boldsymbol{x}}\ \boldsymbol{p})$
5. $[\![\lambda y.M]\!]^k_{\boldsymbol{x}} = \lambda \boldsymbol{p}.\lambda p_y.\ [\![M]\!]^k_{\boldsymbol{x}y} \boldsymbol{p} p_y$

Note that λ symbol on the right hand side of the equality is the semantic symbol used to denote a relevant function, and not a part of the syntax while the sequence $\boldsymbol{p}$ denote a sequence of parameters $p_1, \ldots, p_n$ ranging respectively in $\mathcal{D}^k_{A_1}, \ldots, \mathcal{D}^k_{A_n}$.

Lemma 9 ensures the existence of the meaning of Y in $\mathcal{D}^k$. With this at hand, the next lemma provides all the other facts necessary to show that the meaning of a term with respect to $\boldsymbol{x}$ is always an element of the model.

Lemma 10. *For every sequence of types $\boldsymbol{A} = A_1 \ldots A_n$ and every types B, C we have the following:*

- *For every constant $p \in \mathcal{D}^k_B$ the constant function $f_p : A_1 \to \cdots \to A_n \to B$ belongs to $\mathcal{K}$.*
- *For $i = 1, \ldots, n$, the projection $\pi_i : A_1 \to \ldots A_n \to A_i$ belongs to $\mathcal{K}$.*
- *If $f : \boldsymbol{A} \to (B \to C)$ and $g : \boldsymbol{A} \to B$ are in $\mathcal{D}^k$ then $\lambda \boldsymbol{p}.f\boldsymbol{p}(g\boldsymbol{p}) : \boldsymbol{A} \to C$ is in $\mathcal{D}^k$.*

Proof. For the first item we take $p \in \mathcal{D}^k_B$ and show that the constant function $f_p : A \to B$ belongs to $\mathcal{D}^k_{A \to B}$. For $k = 0$ this is clear. For $k > 0$ we take $p^{\downarrow}$ and consider the constant function $f_{p^{\downarrow}} \in \mathcal{D}^{k-1}_p$. We have $(f_{p^{\downarrow}}, f_p) \in \mathcal{L}^k_{A \to B}$ since $(p^{\downarrow}, p) \in \mathcal{L}^k_B$ by Lemma 1. So $f_p \in \mathcal{D}^k_{A \to B}$.

The (easy) proofs for the second and the third items follow the same kind of reasoning. □

These observations allow us to conclude that $\mathcal{D}^k$ is indeed a model of the λY-calculus, that is:

1. for every term M of type A and every valuation ν ranging of the free variables of M, $[\![M]\!]^k_\nu$ is in $\mathcal{D}^k_A$,
2. given two terms M and N of type A, if $M =_{\beta\delta} N$, then for every valuation ν, $[\![M]\!]^k_\nu = [\![N]\!]^k_\nu$.

Moreover the fundamental Lemma on logical relation has the following consequence.

Lemma 11. *For every $k > 0$, every term M and valuation ν into $\mathcal{D}^{k-1}$ we have $([\![M]\!]^{k-1}_\nu, [\![M]\!]^k_{\nu^{\uparrow_\wedge}}) \in \mathcal{L}^k$ and $([\![M]\!]^{k-1}_\nu, [\![M]\!]^k_{\nu^{\uparrow_\vee}}) \in \mathcal{L}^k$; where $\nu^{\uparrow_\wedge}$ and $\nu^{\uparrow_\vee}$ are as expected.*

B Correctness and completeness of the model

In this part of the appendix, we prove Theorem 3.

Fix a weak alternating automaton $\mathcal{A} = \langle Q, \Sigma, \delta, \rho \rangle$ where Q is a set of states Σ is the alphabet, $\delta : Q \times \Sigma \to \mathcal{P}(\mathcal{P}(Q) \times \mathcal{P}(Q))$ is a transition function, and $\rho : Q \to \mathbb{N}$ a ranking function. Recall that weak means that the states in a transition for a state q have ranks at most $\rho(q)$, in other words, for every $(S_0, S_1) \in \delta(q, a)$, $S_0, S_1 \subseteq Q_{\leq \rho(q)}$.

Recall that the automaton $\mathcal{A}$ accepts a tree t from a state $q \in Q$, if Eve has a winning strategy in the following game between Eve and Adam. The fist turn of a play starts in the root of t with a state q_0. Eve's move in the turn consists of choosing a pair $(S_0, S_1) \in \delta(q, a)$; where a is the label of the root of t. Then Adam chooses S_i (for $i = 0, 1$) and a state $q_1 \in S_i$. The next turn begins in the i-th child of the root and with the the state q_1. The result after infinitely many turns of the play is an infinite sequence of states $q_0, q_1, \ldots$. Eve wins such a play if the rank of all but finitely many states in this sequence is even. Observe that due to the weakness of $\mathcal{A}$ the rank of states in the sequence can never increase, so it eventually stabilizes at some value.

There is one more rule concerning leaves. We need to define transition function for leaves. Leaves are always labeled with Ω and Eve wins in such node iff the current state has even rank.

Recall also that, without a loss of generality, we assume that δ is monotone, i.e. if $(S_0, S_1) \in \delta(q, a)$ and $S_0 \subseteq S_0'$ and $S_1 \subseteq S_1'$ then $(S_0', S_q') \in \delta(q, a)$, and that the semantics of constants is:

$$[\![a]\!]^k(S_0, S_1) = \{q : (S_0, S_1) \in \delta(q, a)\}$$

Notice that, by our assumption about monotonicity of δ, this function is monotone. Let

$$\mathcal{A}(M) = \{q \in Q : \mathcal{A} \text{ accepts } BT(M) \text{ from } q\}$$

be the set of states from which $\mathcal{A}$ accepts the tree $BT(M)$.

We want to show that our model $\mathcal{D}^m$ can calculate $\mathcal{A}(M)$; here m is the maximal value of the rank function of $\mathcal{A}$. The following theorem states a slightly more general fact.

Theorem 3 *For every closed term M of type 0, for every $0 \leq k \leq m$ we have $[\![M]\!]^k = \mathcal{A}(M) \cap Q_{\leq k}$.*

For $k = 0$ the model $\mathcal{D}^0$ is just the GFP model over Q_0. Moreover $\mathcal{A}$ restricted to the states in Q_0 is an automaton with trivial conditions. The theorem follows from Theorem 1.

For the induction step take even $k > 0$. The case of odd k is similar and we will not present it here.

The next lemma shows left to right inclusion of Theorem 3.

Lemma 12. $[\![M]\!]^k \subseteq \mathcal{A}(M)$

Proof. We take $q \in [\![M]\!]^k$ and describe a winning strategy for *Eve* on $BT(M)$ from q. If the rank of $q < k$ then such a strategy exist by the induction assumption. So we suppose that $\rho(q) = k$.

If M does not have a head normal form then $BT(M)$ consists just of the root ε labelled with Ω. Then Eve wins by definition of the game since k is even.

Suppose then that M has a head normal form aM_0M_1. As $[\![M]\!]^k = [\![aM_0M_1]\!]^k$ we have $q \in [\![aM_0M_1]\!]^k$. By the semantics of a we know that $([\![M_0]\!]^k, [\![M_1]\!]^k) \in \delta(q, a)$. The strategy of Eve is to choose $([\![M_0]\!]^k, [\![M_1]\!]^k)$. Suppose Adam then selects $q_1 \in [\![M_1]\!]^k$. If $\rho(q_1) < k$ then Eve has a winning strategy by induction hypothesis. Otherwise, if $\rho(q_1) = k$ we repeat the reasoning.

This strategy is winning for Eve since a play either stays in states of even rank k or switches to a play following a winning strategy for smaller ranks. □

It remains to show that $\mathcal{A}(M) \cap Q_{\leq k} \subseteq [\![M]\!]^k$. For this we will define one logical relation between $\mathcal{D}^k$ and the syntactic model of λY and show a couple of lemmas.

Definition 4. *We define a logical relation between the model $\mathcal{D}^k$ and closed terms*

$$\begin{aligned} R_0 &= \{(P, M) : \mathcal{A}(M) \subseteq P \cap Q_{\leq k} \text{ and } P \cap Q_{\leq k-1} = [\![M]\!]^{k-1}\} \ , \\ R_{A\to B} &= \{(f, M) : \forall_{(g,N)\in R_A}.\ (f(g), MN) \in R_\beta\} \ . \end{aligned}$$

The first observation follows immediately from the fact that R is a logical relation.

Lemma 13. *If $M =_{\beta\delta} N$ and $(f, M) \in R_A$ then $(f, N) \in R_A$.*

Next we introduce an equivalence relation on $\mathcal{D}^k$. The point is that there are many elements in $\mathcal{D}^k$ that are not meanings of terms. So we say that two functions are equivalent if they behave the same on meanings of terms (heridetary).

Definition 5. *We define a family of binary relations $\approx_A$ on $\mathcal{D}^k_A$:*

$$\begin{aligned} d \approx_0 e \quad & \text{iff } d = e \\ f \approx_{B\to C} g \quad & \text{iff for all } d \in \mathrm{Dom}(R_B),\ f(d) \approx_C g(d) \end{aligned}$$

where $\mathrm{Dom}(R_B) = \{d : \exists N.\ (d, N) \in R_B\}$ is the set of elements in relation with some term.

Lemma 14. *For every type A relation $\approx_A$ is an equivalence relation.*

Lemma 15. *If $f \approx_A ([\![M]\!]^{k-1})^{\uparrow\vee}$ then $(f, M) \in R_A$.*

Proof. If M is of type 0 then we can use the induction hypothesis given by Theorem 3. From $[\![M]\!]^{k-1} = \mathcal{A}(M) \cap Q_{\leq k-1}$ the lemma is immediate by definition of R_0.

Consider now M of type $B \to C$. Take some $(h, N) \in R_B$. Since $f \approx_A (\llbracket M \rrbracket^{k-1})^{\uparrow\vee}$ we get $f(h) \approx_C (\llbracket M \rrbracket^{k-1})^{\uparrow\vee}(\llbracket N \rrbracket^{k-1})^{\uparrow\vee}$. At this point let us observe that $g^{\uparrow\vee}(d^{\uparrow\vee}) = (g(d))^{\uparrow\vee}$. Indeed, $g^{\uparrow\vee}(d^{\uparrow\vee}) = (g((d^{\uparrow\vee})^{\downarrow}))^{\uparrow\vee}$ by Corollary 2; and $(d^{\uparrow\vee})^{\downarrow} = d$ by Lemma 1. Using this observation we get $f(h) \approx_C (\llbracket M \rrbracket^{k-1}\llbracket N \rrbracket^{k-1})^{\uparrow\vee}$. By induction hypothesis we obtain $(f(h), MN) \in R_C$. This shows $(f, M) \in R_{B\to C}$. □

Lemma 16. *Let v be a valuation, and σ a substitution of closed terms such that $(v(x^A), \sigma(x^A)) \in R_A$ for every variable x^A in the domain of σ. For every term M of a type A we have $(\llbracket M \rrbracket^k_v, M.\sigma) \in R_A$.*

Proof. The proof is by induction on the structure of M.

If M is a variable then the proof is immediate.

If M is a constant a then we show that $(\llbracket a \rrbracket, a) \in R_{0\to 0\to 0}$. For this we take $(S_0, N_0), (S_1, N_1) \in R_0$, and we show $(\llbracket a \rrbracket^k(S_0, S_1), aN_0N_1) \in R_0$. Take $q \in \mathcal{A}(aN_0N_1)$. If the rank of q is smaller than k then $S_0, S_1 \subseteq Q_{\leq k-1}$ and we conclude by the outermost induction assumption saying that $\llbracket aN_0N_1 \rrbracket^{k-1} = \mathcal{A}(aN_0N_1) \cap Q_{\leq k-1}$. If the rank of q is k, then we look at Eve's winning strategy in the acceptance game from q on $BT(aN_0N_1)$. In the first round of this game she chooses some $(T_0, T_1) \in \delta(a, q)$. Since her strategy is winning we have $T_0 \subseteq \mathcal{A}(N_0)$ and $T_1 \subseteq \mathcal{A}(N_1)$. So $q \in \llbracket a \rrbracket(T_0, T_1)$. From the definition of R_0 we get $\mathcal{A}(N_0) \subseteq S_0$ and $\mathcal{A}(N_1) \subseteq S_1$. By monotonicity we get the desired $q \in \llbracket a \rrbracket(S_0, S_1)$.

If M is an application NP then the conclusion is immediate from the definition of R_A.

If M is an abstraction $\lambda x.\ N : B \to C$, then we take $(g, P) \in R_B$. By induction hypothesis $(\llbracket N \rrbracket^k_{v[g/x]}, N.\sigma[P/x]) \in R_C$. So $(\llbracket M \rrbracket^k_v(g), MP) \in R_C$ by Lemma 13.

If $M = Y^{(A\to A)\to A}$. Take $(f, P) \in R_{A\to A}$. By Lemma 15 $((\mathrm{fix}^{k-1}_A(f^{\downarrow}))^{\uparrow\vee}, YP) \in R_A$. Then, using Lemma 13, $(f((\mathrm{fix}^{k-1}_\alpha(f^{\downarrow}))^{\uparrow\vee}), YP) \in R_A$, and consequently $(f^i((\mathrm{fix}^{k-1}_A(f^{\downarrow}))^{\uparrow\vee}), YP) \in R_\alpha$ for all $i \geq 0$. Since the sequence of $f^i((\mathrm{fix}^{k-1}_A(f^{\downarrow}))^{\uparrow\vee})$ is decreasing, it reaches the fixpoint $\mathrm{fix}^k_A(f)$ in a finite number of steps and $(\mathrm{fix}^k_A(f), YP)$ is in R_A. As (f, P) is an arbitrary element of $R_{A\to A}$, this shows that (fix^k_A, Y) is in $R_{(A\to A)\to A}$. □

C From the model to type systems

We here give the properties of types and their interpretation in the model. We split the proof of Lemma 3 into several lemmas.

Lemma 17. *For every type A, $S \subseteq \mathrm{Types}_A$ and $k \leq m$ we have $\llbracket S \rrbracket^k = \llbracket S \cap \mathrm{Types}^k_A \rrbracket^k$.*

Proof. Since $\llbracket S \rrbracket^l = \bigvee\{\llbracket t \rrbracket^l : t \in S\}$ it is sufficient to show that $\llbracket t \rrbracket^k = \bot^k_A$ for $t \notin \mathrm{Types}^k_A$. We do it by induction on the type A.

Suppose that $t \in \text{types}^l_A$ for $l > k$. For type 0 it follows directly from the definition that $[\![t]\!]^k = \emptyset$. For A of the form $B \to C$ we know that t is of the form $T \to s$ with $T \subseteq \text{Types}^k_B$ and $s \in \text{types}^l_C$. By induction assumption $[\![s]\!]^k = \bot^k_C$. We get $[\![T \to s]\!]^k = [\![T]\!]^k \rightharpoonup^k [\![s]\!]^k = [\![T]\!]^k \rightharpoonup^k \bot_C = \bot^k_{B\to C}$. □

Lemma 18. *For every $k < m$ and A, if $S \subseteq \text{Types}_A$, then $[\![S \cap \text{Types}^k_A]\!]^{k+1} = ([\![S]\!]^k)^{\uparrow\wedge}$.*

Proof. We prove the result only for elements of types^k_A as the more general one is a direct consequence of that particular case. The proof is by induction on A. The base case is obvious. For A of the form $B \to C$ we have $[\![T \to s]\!]^{k+1}$ is by definition $[\![T]\!]^{k+1} \rightharpoonup^{k+1} [\![s]\!]^{k+1}$ which by induction hypothesis is $([\![T]\!]^k)^{\uparrow\wedge} \rightharpoonup^{k+1} ([\![s]\!]^k)^{\uparrow\wedge}$. We will be done if we show that for every $f \in \mathcal{D}^k_B$ and $g \in \mathcal{D}^k_C$:

$$f^{\uparrow\wedge} \rightharpoonup^{k+1} g^{\uparrow\wedge} = (f \rightharpoonup^k g)^{\uparrow\wedge} \ .$$

Given e in $\mathcal{D}^{k+1}_B$, by Corollary 2, we have $(f \rightharpoonup^k g)^{\uparrow\wedge}(e)$ is equal to $((f \rightharpoonup^k g)(e^{\downarrow}))^{\uparrow\wedge}$, and therefore:

$$(f \rightharpoonup^k g)^{\uparrow\wedge}(e) = \begin{cases} g^{\uparrow\wedge} & \text{if } f \leq e^{\downarrow}, \\ \bot^{k+1}_C & \text{otherwise.} \end{cases}$$

Since $f \leq e^{\downarrow}$ iff $f^{\uparrow\wedge} \leq e$, by Corollary 1, this proves the desired equality. □

Lemma 4 *For every $k \leq m$ and every type A. For every f in $\mathcal{D}^k_A$ there is $S \subseteq \text{Types}^k_A$ so that $[\![S]\!]^k = f$, and there is $S' \subseteq \text{types}^k_A$ so that $[\![S']\!]^k = \overline{f}$.*

Proof. We proceed by induction on k.

The case where $k = 0$ has been proved in [4].

For the case $k > 0$, as we have seen with Lemma 2, that $f = (f^{\downarrow})^{\uparrow\wedge} \vee \overline{f}$. From the induction hypothesis there is $S_1 \subseteq \text{Types}^{k-1}_A$ such that $[\![S_1]\!]^{k-1} = f^{\downarrow}$. By Lemma 18 we get $[\![S_1]\!]^k = (f^{\downarrow})^{\uparrow\wedge}$.

It remains to describe $\overline{f}$ with types from $\text{types}^k_{B\to C}$. Take $d \in \mathcal{D}^k_B$ and recall that $\overline{f}(d) = \overline{f(d)}$. By induction hypothesis we have $S_d \subseteq \text{Types}^k_B$ and $S_{\overline{f(d)}} \subseteq \text{types}^k_C$ such that $[\![S_d]\!]^k = d$ and $[\![S_{\overline{f(d)}}]\!]^k = f(d)$. So the set of types $S_d \to S_{\overline{f(d)}}$ is included in $\text{types}^k_{B\to C}$ and $[\![S_d \to S_{\overline{f(d)}}]\!] = d \rightharpoonup^k \overline{f}(d)$. It remains to take $S_2 = \bigcup\{S_d | d \in \mathcal{D}^k_B\}$. We can conclude that $S_2 \subseteq \text{types}^k_A$ and $[\![S_2]\!]^k = \overline{f}$. Therefore $[\![S_1 \cup S_2]\!]^k = f$. □

Theorem 23 *For each k, A, S and T in Types^k_A, we have that $[\![S]\!]^k \leq [\![T]\!]^k$ iff $S \sqsubseteq_A T$.*

Proof. This Theorem is a consequence of the fact that for each k, $\mathcal{D}_k$ can be embedded in the monotone model generated by $\mathcal{P}(Q_{\leq k})$ and that according to [18], the ordering on intersection types simulate the one in monotone models. □

The main Theorem 4 is proved by two lemmas.

Lemma 19. *If $\Gamma \vdash M \geq S$ is derivable, then for every $k \leq m$: $[\![M]\!]^k_{[\![\Gamma]\!]^k} \geq [\![S]\!]^k$.*

Proof. This proof is done by a simple induction on the structure of the derivation of $\Gamma \vdash M \geq S$. For most of the rules, the conclusion follows immediately from the induction hypothesis (using Lemmas 23 and 6). We shall only treat here the case of the rules Y *odd* and Y *even*.

In the case of Y *odd*, when we derive $\Gamma \vdash Yx.M \geq S(T)$ from $\Gamma \vdash \lambda x.M \geq S$ and $\Gamma \vdash Yx.M \geq T$ with $S, T \in \mathrm{Types}^{2l+1}_A$, the induction hypothesis gives that for every k, $[\![\lambda x.M]\!]^k_{[\![\Gamma]\!]^k} \geq [\![S]\!]^k$ and $[\![Yx.M]\!]^k_{[\![\Gamma]\!]^k} \geq [\![T]\!]^k$. Therefore $[\![Yx.M]\!]^k_{[\![\Gamma]\!]^k} = [\![(\lambda x.M)(Yx.M)]\!]^k_{[\![\Gamma]\!]^k} \geq [\![S]\!]^k([\![T]\!]^k) = [\![S(T)]\!]^k$, using Lemma 6.

In the case of Y *even* we consider the case $k = 2l$. Let ν_{k-1} stand for $[\![\Gamma]\!]^{k-1}$ and ν_k for $[\![\Gamma]\!]^k$.

By induction hypothesis we have $[\![Yx.M]\!]^{k-1}_{\nu_{k-1}} \geq [\![T]\!]^{k-1}$. Since Lemma 11 implies $([\![Yx.M]\!]^{k-1}_{\nu_{k-1}}, [\![Yx.M]\!]^k_{\nu_k}) \in \mathcal{L}^k$, we have $[\![Yx.M]\!]^k_{\nu_k} \geq ([\![T]\!]^{k-1})^{\uparrow\wedge}$ by Lemma 1. By Lemma 18 we know $([\![T]\!]^{k-1})^{\uparrow\wedge} = [\![T]\!]^k$. In consequence we have

$$[\![\lambda x.M]\!]^k_{\nu_k}([\![T]\!]^k) \geq [\![T]\!]^k.$$

Also by induction hypothesis we have $[\![\lambda x.M]\!]^k_{\nu_k} \geq [\![(S \cup T) \to S]\!]^k$. This means $[\![\lambda x.M]\!]^k_{\nu_k}([\![S \cup T]\!]^k) \geq [\![S]\!]^k$. Put together with what we have concluded about $[\![T]\!]^k$ we get

$$[\![\lambda x.M]\!]^k_{\nu_k}([\![S \cup T]\!]^k) \geq [\![S \cup T]\!]^k.$$

Now we use Lemma 4 telling us that

$$[\![Yx.M]\!]^k_{\nu_k} = \bigvee \{d \mid [\![\lambda x.M]\!]^k(d) \geq d \text{ and } d^{\downarrow} = [\![Yx.M]\!]^{k-1}_{\nu_{2k-1}}\} \ .$$

This gives us immediately the desired $[\![Yx.M]\!]^k_{\nu_k} \geq [\![S \cup T]\!]^k$. □

Lemma 20. *Given a type $S \subseteq \mathrm{Types}^k$, if $[\![M]\!]^k_{[\![\Gamma]\!]^k} \geq [\![S]\!]^k$ then $\Gamma \vdash M \geq S$.*

Proof. This theorem is proved by induction on the pairs (M, k) ordered componentwise. Suppose that the statement is true for M, we are going to show that it is true for $Yx.M$, the other cases are straightforward.

The first observation is that, if $T \to S$ is such that $[\![\lambda x.M]\!]^k_{[\![\Gamma]\!]^k} \geq [\![T \to S]\!]^k$, then $\Gamma \vdash \lambda x.M \geq T \to S$ is derivable. Indeed, since, letting $\nu = [\![\Gamma, x \geq T]\!]^k$), if $[\![M]\!]^k_\nu \geq [\![S]\!]^k$ holds then, $\Gamma, x \geq T \vdash M \geq S$ is derivable by induction hypothesis. So $\Gamma \vdash \lambda x.M \geq T \to S$ is derivable.

There are now two cases depending on the parity of k. First let us assume that k is even. Suppose $[\![Yx.M]\!]^k_{[\![\Gamma]\!]^k} = [\![S \cup T]\!]^k$ where $S \subseteq \mathrm{types}^k$ and $T \subseteq \mathrm{Types}^{k-1}$. Lemma 4 guaranties the existence of such S and T as every element of $\mathcal{D}^k$ is expressible by a set of types. We have $[\![\lambda x.M]\!]^k_{[\![\Gamma]\!]^k} \geq [\![S \cup T]\!]^k \to^k [\![S \cup T]\!]^k$. By

the above we get that $\Gamma \vdash \lambda x.M \geq (S \cup T) \to (S \cup T)$ is derivable and thus $\Gamma \vdash \lambda x.M \geq (S \cup T) \to S$ is also derivable. We also have $[\![Yx.M]\!]^{k-1} \geq [\![T]\!]^{k-1}$ which gives that $\Gamma \vdash Yx.M \geq T$ is derivable. This allows us to derive $\Gamma \vdash Yx.M \geq S \cup T$. Using the subsumption rule and Lemma 23 every other valid judgment $\Gamma \vdash Yx.M \geq U$ is derivable.

Now consider the case where k is odd. Suppose $[\![Yx.M]\!]^k_{[\![\Gamma]\!]^k} = [\![S \cup T]\!]_k$ with $T \subseteq \text{Types}^{k-1}$ and $S \subseteq \text{types}^k$. By induction hypothesis on k, we have that $\Gamma \vdash M \geq T$ is derivable. Take $d = [\![\lambda x.M]\!]^k_{[\![\Gamma]\!]_k}$. Lemma 4 guarantees us a set of types $U \subseteq \text{Types}^k$ such that $[\![U]\!]^k = d$. By the observation we have made above, there is a derivation of $\Gamma \vdash \lambda x.M \geq U$. Then iteratively using the rule Y *odd* we compute the least fixpoint by letting $U^0(T) = T$ and $U^{n+1}(T) = U(U^n(T))$. $\square$

Concerning the dual system, duality on lattices makes it possible to derive directly its properties from the previous system. Nevertheless, for the sake of completeness, we included the main properties of the dual system here (including a complete version of its derivation system given Figure 4).

Similarly to the definition of $[\![\cdot]\!]^k$, we write $\langle\!\langle S \rangle\!\rangle^k$ for $\bigwedge\{\langle\!\langle s \rangle\!\rangle^k : s \in S\}$ and we also have that $\langle\!\langle T \to S \rangle\!\rangle^k = \langle\!\langle T \rangle\!\rangle^k \rightharpoonup^k \langle\!\langle S \rangle\!\rangle^k$. Simply by dualizing the proof of the corresponding lemmas for $[\![\cdot]\!]^k$, we can establish the following results:

Lemma 21. *For every type A, if $S \subseteq \text{Types}_A$ and $k < m$ we have: have that $\langle\!\langle S \rangle\!\rangle^k = \langle\!\langle S \cap \text{Types}^k_A \rangle\!\rangle$, $\langle\!\langle S \cap \text{Types}^k_A \rangle\!\rangle^{k+1} = (\langle\!\langle S \rangle\!\rangle^k)^{\uparrow\vee}$ and $\langle\!\langle S \rangle\!\rangle^k = (\langle\!\langle S \rangle\!\rangle^{k+1})^{\downarrow}$.*

Lemma 22. *For every k, every A, and every f in $\mathcal{D}^k_A$, there is S in Types^k_A so that $\langle\!\langle S \rangle\!\rangle^k = f$.*

Lemma 23. *For each k, A, S and T in Types^k_A, we have that $\langle\!\langle S \rangle\!\rangle^k \geq \langle\!\langle T \rangle\!\rangle^k$ iff $S \sqsupseteq_A T$.*

We now define $S(T)$ to be $\{s : U \to s \in S \wedge U \sqsupseteq T\}$. We then have:

Lemma 24. *For every k, A and B, if $S \subseteq \text{Types}_{A \to B}$ and $T \subseteq \text{Types}A$, then $\langle\!\langle S(T) \rangle\!\rangle^k = \langle\!\langle S \rangle\!\rangle^k(\langle\!\langle T \rangle\!\rangle^k)$.*

$$\frac{S \subseteq T \subseteq Q}{S \sqsupseteq_0 T} \qquad \frac{\forall s \in S, \exists t \in T, s \sqsupseteq_A t}{S \sqsupseteq_A T} \qquad \frac{T \sqsupseteq_A S \quad s \sqsupseteq_B t}{S \to s \sqsupseteq_{A\to B} T \to t}$$

$$\frac{}{\Gamma, x \not\geq S \vdash x \not\geq S} \qquad \frac{\Gamma \vdash M \not\geq S \quad \Gamma \vdash M \not\geq T}{\Gamma \vdash M \not\geq S \cup T} \qquad \frac{\Gamma \vdash M \not\geq S \quad T \sqsupseteq S}{\Gamma \vdash M \not\geq T}$$

$$\frac{\forall (S_1, S_2) \in \delta(a, q), (T_1 \cap S_1) \cup (T_2 \cap S_2) \neq \emptyset}{\Gamma \vdash a \not\geq \{T_1 \to T_2 \to q\}}$$

$$\frac{\Gamma \vdash M \not\geq S \quad \Gamma \vdash N \not\geq T}{\Gamma \vdash MN \not\geq S(T)} \qquad \frac{S \in \mathrm{Types}^k,\ T \subseteq \mathrm{types}^k \quad \Gamma, x \not\geq S \vdash M \not\geq T}{\Gamma \vdash \lambda x.M \not\geq S \to T}$$

$$\frac{S \subseteq \mathrm{types}_A^{2k+1}, \quad T \in \mathrm{Types}_A^{2k}, \quad \Gamma \vdash \lambda x.M \not\geq (S \cup T) \to S \quad \Gamma \vdash Yx.M \not\geq T}{\Gamma \vdash Yx.M \not\geq S \cup T}\ Y\ odd$$

$$\frac{S, T \in \mathrm{Types}_A^{2k}, \quad \Gamma \vdash (\lambda x.M) \not\geq S \quad \Gamma \vdash (Yx.M) \not\geq T}{\Gamma \vdash Yx.M \not\geq S(T)}\ Y\ even$$

Fig. 4. Complete dual type system