Models of λ-Calculus and the Weak MSO Logic[*]

Paweł Parys[1] and Szymon Toruńczyk[2]

1 **University of Warsaw, Warsaw, Poland**
parys@mimuw.edu.pl
2 **University of Warsaw, Warsaw, Poland**
szymtor@mimuw.edu.pl

Abstract

We study the Weak MSO logic in relationship to infinitary λ-calculus. We show that for every formula φ of Weak MSO there exists a finitary model of infinitary λ-calculus recognizing the set of infinitary λ-terms whose Böhm tree satisfies φ. The model is effective, in the sense that for every λY-term we can effectively compute its value in the model. In particular, given a finite λY-term, one can decide whether the resulting Böhm tree satisfies a given formula of Weak MSO, which is a special case of the result of Ong [16], which concerns unrestricted MSO. The existence of effective models for Weak MSO and MSO was proved earlier by Salvati and Walukiewicz [19, 20] but our proof uses a different method, as it does not involve automata, but works directly with logics.

1998 ACM Subject Classification F.1.1 Models of Computation

Keywords and phrases typed λ-calculus, models, Weak MSO logic

Digital Object Identifier 10.4230/LIPIcs.CSL.2016.11

1 Introduction

In this paper, we study the model-checking problem on infinite trees generated by finite terms. More precisely, we consider simply typed λY-calculus – an extension of simply typed λ-calculus by a fixpoint operator Y. Any term K of simply typed λY-calculus generates an infinite tree, called the Böhm tree $BT(K)$, which reflects the control flow of the program corresponding to the term. We illustrate this with an example borrowed from [20], depicted in Figure 1.

Trees generated by terms of simply typed λY-calculus can be used to faithfully represent the workflow of programs in a language with higher-order functions. Traditionally, higher-order recursion schemes are used for this purpose [6, 12, 16, 13]; this formalism is equivalent to simply typed λY-calculus [18], and the translation between them is rather straightforward. Collapsible pushdown systems [9] and ordered tree-pushdown systems [5] are other equivalent formalisms.

The model-checking problem for λY-calculus is the following: given a closed λY-term K of the ground type, and a formula φ talking about trees, decide whether $BT(K) \models \varphi$. In this paper, we consider this problem, where the formula φ is a formula of Weak MSO logic. This logic extends first-order logic by allowing to quantify existentially and universally over finite sets of vertices. For example, the formula $\exists_{\mathsf{fin}} X.\forall x.(x \in X \iff a(x))$ holds in a tree t iff t has finitely many nodes labeled with a.

[*] Work supported by the National Science Center (decision DEC-2012/07/D/ST6/02443).

$$Factorial(x) \equiv \texttt{if x = 0 then 1 else x} \cdot Factorial(\texttt{x} - 1)$$
$$Fct \equiv Y(\lambda f.(\lambda x.\texttt{if } (\texttt{isZero } x)\ 1\ (x * f(x-1))))$$

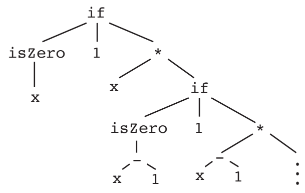

Figure 1 The factorial function, the corresponding λY-term, and its Böhm tree.

Weak MSO logic can capture many natural properties important in verification, such as safety or liveness. It is known from [16] that model-checking for a strictly larger logic is decidable, namely, for MSO logic, where quantification is not restricted to finite sets only. The same was later shown in [9, 13, 18, 23] using different methods.

The main result of our paper states that for every sentence φ of Weak MSO one can construct an effective and finitary *model* of simply typed λY-calculus, recognizing the set of trees in which φ is true. Roughly, this means that to every closed term K of λY-calculus we assign its value $[\![K]\!]_\varphi$, that can be effectively computed from K and φ. This assignment is such that only finitely many elements are used as values of terms of every fixed type. It also preserves composition, i.e., for the composition $K\,M$ of two terms, we have $[\![K\,M]\!]_\varphi = [\![K]\!]_\varphi\,[\![M]\!]_\varphi$, for appropriately defined composition $[\![K]\!]_\varphi\,[\![M]\!]_\varphi$ of elements of the model. Moreover, when K is of the ground type, we can determine whether $BT(K) \models \varphi$ holds seeing only $[\![K]\!]_\varphi$. A formal definition of a model is given in Section 2.

▶ **Theorem 1.1.** *For every sentence of φ of* Weak MSO *there exists an effective and finitary model of simply typed λY-calculus, that recognizes the set of trees in which φ is true.*

We remark that the term "model of λ-calculus" has various meanings in the literature [15, 2, 14], and the notion of model roughly described above is sometimes called an *applicative structure*. Our applicative structure has additional properties, namely that the value of a λY-term can be computed in a compositional way, from the values of its subterms.

The above result was proved earlier in [20]. In [19] and [7] analogous result was proved in full generality for MSO, and in [1] and [21] for properties definable by so-called TAC automata that are even less expressive than the Weak MSO logic. In this paper, we give yet another proof of this result for Weak MSO. Our approach is different than that of the cited papers, in that those papers work with and exploit the structure of parity automata corresponding to the logic, whereas we work directly with formulae.

It is argued in [21] that having a model has several other virtues in addition to providing decidability of the model-checking problem. In particular it allows to obtain the reflection property [4, 8] and the transfer theorem [17].

The paper [20], besides proving Theorem 1.1, provides a type system, such that typing a term using this system is equivalent to the Böhm tree being accepted by the fixed weak alternating automaton. The approach to model checking of Böhm trees using type systems has many advantages, and appears earlier in [13, 23, 11].

2 Preliminaries

Trees. Let Σ be a *ranked alphabet*, i.e., a set of symbols together with a rank function *rank* assigning a nonnegative integer to each of the symbols. Apart from Σ, we have a special symbol $\perp \notin \Sigma$ of rank 0, used to label a „missing part" of a tree. A Σ-*labeled* tree is a tree which is rooted (there is a distinguished root node which is the ancestor of every node in the tree), node-labeled (every node has a label from $\Sigma \cup \{\bot\}$), ranked (a node with label of rank n has exactly n children), and ordered (the children of a node of rank n are numbered from 1 to n).

Weak MSO. The Weak MSO logic is a restriction of the MSO logic, in which set quantifiers range only over finite sets. For technical convenience, we use a variant of Weak MSO in which there are no first-order variables, and where set variables do not contain nodes labeled by $\bot$. It is easy to translate a formula from any standard syntax of Weak MSO to ours (at least when the alphabet Σ is finite). In the syntax of Weak MSO we have the following constructions:

$$\varphi ::= a(X) \mid X <_i Y \mid X \subseteq Y \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi' \mid \exists_{\mathsf{fin}} X.\varphi' \qquad \text{where } a \in \Sigma, i \in \mathbb{N}_+.$$

We evaluate formulae of Weak MSO in Σ-labeled trees. Set variables are interpreted as sets of nodes labeled by elements of Σ, and the semantics of formulae is defined as follows:

- $a(X)$ holds iff every node in X is labeled by a,
- $X <_i Y$ holds iff every node in Y is a (not necessarily proper) descendant of the i-th child of every node in X,
- $X \subseteq Y$, $\varphi_1 \wedge \varphi_2$, and $\neg\varphi'$ are defined as expected, and
- $\exists_{\mathsf{fin}} X.\varphi'$ holds iff φ' holds for some finite set X of nodes labeled by elements of Σ.

Infinitary λ-calculus. We consider infinitary, simply typed λ-calculus. In particular, each term has an associated type.

The set of *types* is constructed from a unique ground type o using a binary operation $\to$. Thus o is a type and if α, β are types, so is $(\alpha \to \beta)$. The order of a type is defined by: $\mathit{ord}(o) = 0$, and $\mathit{ord}(\alpha \to \beta) = \max(1 + \mathit{ord}(\alpha), \mathit{ord}(\beta))$. By convention, $\to$ associates to the right, i.e., $\alpha \to \beta \to \gamma$ is understood as $\alpha \to (\beta \to \gamma)$. Every type α can be uniquely written as $\alpha_1 \to \alpha_2 \to \ldots \to \alpha_n \to o$. The type $\underbrace{o \to \cdots \to o}_{k} \to o$ is denoted $o^k \to o$, where $o^0 \to o$ is simply o.

Infinitary λ-terms (or just *terms*) are defined by coinduction, according to the following rules:

- For each symbol $a \in \Sigma$ of rank r, $a^{o^r \to o}$ is a term.
- For each type α there are infinitely many variables $x^\alpha, y^\alpha, z^\alpha, \ldots$; each of them is a term.
- If $K^{\alpha\to\beta}$ and M^α are terms, then $(K^{\alpha\to\beta}\, M^\alpha)^\beta$ is a term.
- If K^β is a term and x^α is a variable, then $(\lambda x^\alpha.K^\beta)^{\alpha\to\beta}$ is a term.

We naturally identify two terms if they are α-equivalent. We often omit the type annotations of terms, but we keep in mind that every term has a fixed type. The set $FV(K)$ of free variables of a term K is defined as expected. A term K is *closed* if $FV(K) = \emptyset$. We denote terms by capital roman letters, e.g. $K, M, N, P, Q, \ldots$

λY-calculus. The syntax of λY-calculus is the same as of finite λ-calculus, extended by symbols $Y^{(\alpha\to\alpha)\to\alpha}$, for each type α. A term of λY-calculus is seen as a term of infinitary λ-calculus, by substituting each symbol $Y^{(\alpha\to\alpha)\to\alpha}$ by the unique term Y such that Y is equal to $\lambda M^{\alpha\to\alpha}.M\,(Y\,M)$. In this way, we view λY-calculus as a fragment of infinitary λ-calculus.

Trees Generated by Terms. Let K be a closed (infinitary) term of type o. The tree t *generated* by K is constructed by coinduction, as follows: if there is a sequence of β-reductions from K to a term of the form $a\,K_1\,\dots\,K_r$, where a is a symbol (of rank r), then the root of the tree t has label a, and, for $i \in \{1,\dots,r\}$, the i-th child of the root of t is the root of the tree generated by K_i. If there is no sequence of β-reductions from K to a term of the above form, the tree generated by K consists of a single node labeled by $\bot$.

The tree generated by K is also called the *Böhm tree* of K, and denoted $BT(K)$.

Models. In this paper, we choose the following definition of a model. An *applicative structure* D consists of a set $D[\alpha]$ for each type α, and of an application operation that to all elements $\sigma \in D[\alpha \to \beta]$ and $\tau \in D[\alpha]$ assigns an element $(\sigma\,\tau) \in D[\beta]$. A D*-valuation* is a partial function v that maps some variables of λ-calculus to elements of the applicative structure, so that $v(x^\alpha) \in D[\alpha]$.

A *model* $\mathcal{M}$ of infinitary λ-calculus consists of an applicative structure $D_\mathcal{M}$, called the *domain*, and a mapping that to each term K^α and to each $D_\mathcal{M}$-valuation v with $\mathrm{dom}(v) \supseteq FV(K)$ assigns an element $[\![K]\!]^v_\mathcal{M} \in D_\mathcal{M}[\alpha]$ so that

M1. $[\![K]\!]^v_\mathcal{M} = [\![K]\!]^w_\mathcal{M}$ whenever $v(x) = w(x)$ for all $x \in FV(K)$,

M2. $[\![K\,M]\!]^v_\mathcal{M} = [\![K]\!]^v_\mathcal{M}\,[\![M]\!]^v_\mathcal{M}$,

M3. $[\![\lambda x.K]\!]^v_\mathcal{M}\,\sigma = [\![K]\!]^{v[x\mapsto\sigma]}_\mathcal{M}$ (informally, $[\![\lambda x.K]\!]^v_\mathcal{M} = \lambda\sigma.[\![K]\!]^{v[x\mapsto\sigma]}_\mathcal{M}$), and

M4. $[\![K[M/x]]\!]^v_\mathcal{M} = [\![K]\!]^{v[x\mapsto[\![M]\!]^v_\mathcal{M}]}_\mathcal{M}$.

When K is closed, we write $[\![K]\!]_\mathcal{M}$ for $[\![K]\!]^v_\mathcal{M}$ (which is independent from v). The following fact follows easily from conditions M2–M4.

▶ **Fact 2.1.** *In every model $\mathcal{M}$ it holds that $[\![K]\!]^v_\mathcal{M} = [\![K']\!]^v_\mathcal{M}$ whenever K and K' are β-equivalent.*

We say that a model $\mathcal{M}$ *recognizes* a set of trees $\mathcal{L}$ if there is a subset F of $D_\mathcal{M}[o]$ such that for every closed term K^o, the tree generated by K belongs to $\mathcal{L}$ if and only if $[\![K]\!]_\mathcal{M} \in F$. We say that a model $\mathcal{M}$ is *finitary* if $D_\mathcal{M}[\alpha]$ is finite for each type α, and furthermore it is *effective*, if for a given λY-term K, one can compute $[\![K]\!]_\mathcal{M}$ (where K is treated as an infinitary λ-term obtained by expanding all Y symbols).

The main result of this paper, Theorem 1.1, states that there exists a finitary, effective model of infinitary λ-calculus recognizing any set of trees definable by a Weak MSO formula. Moreover, we will see that the value of a λY-term can be computed in a compositional way.

3 Phenotypes of Trees

In this section, we define phenotypes. These are objects that characterize properties of trees with respect to a given formula of Weak MSO, and that are compositional.

Let $\mathcal{F}$ be a finite set of variables of Weak MSO. An $\mathcal{F}$*-tree* is a pair $t \otimes v$, where t is a Σ-labeled tree, and v is a valuation of the variables in $\mathcal{F}$ in t. We write $\hat{t}$ to denote $\mathcal{F}$-trees, for some $\mathcal{F}$. If $\hat{t} = t \otimes v$ is a $\mathcal{F}$-tree and x is a node of $\hat{t}$, then the subtree of $\hat{t}$ rooted at x is the $\hat{F}$-tree $s \otimes w$ consisting of the subtree s of t rooted at t and the valuation w which is v restricted to the nodes of s (i.e., for a variable X, $w(X)$ is $v(X)$ intersected with the nodes of s).

Suppose that φ is a formula with free variables contained in $\mathcal{F}$ and $\hat{t} = t \otimes v$ is an $\mathcal{F}$-tree. Then we write $\hat{t} \models \varphi$ if $t, v \models \varphi$. We also define the φ*-phenotype* of $\hat{t} = t \otimes v$, denoted $[\hat{t}]_\varphi$, by induction on the size of the formula φ as follows:

- if φ is of the form $a(X)$ (for some symbol $a \in \Sigma$) or $X \subseteq Y$ then $[\hat{t}]_\varphi$ is the logical value of φ in $\hat{t}$, i.e., *true* if $t, v \models \varphi$ and *false* otherwise,
- if φ is of the form $X <_i Y$, then $[\hat{t}]_\varphi$ is the triple whose first element is the logical value of φ in $\hat{t}$, the second element is *true* if $v(X) = \emptyset$ and *false* otherwise, and the third element analogously for Y,
- if $\varphi = (\psi_1 \wedge \psi_2)$, then $[\hat{t}]_\varphi = ([\hat{t}]_{\psi_1}, [\hat{t}]_{\psi_2})$,
- if $\varphi = (\neg\psi)$, then $[\hat{t}]_\varphi = [\hat{t}]_\psi$, and
- if $\varphi = \exists_{\mathsf{fin}} X.\psi$, then $[\hat{t}]_\varphi = \{[t \otimes w]_\psi \mid w$ is a valuation extending v to $X\}$.

For each φ, let Pht_φ denote the set of all potential φ-phenotypes. Namely, $\mathrm{Pht}_\varphi = \{\mathit{true}, \mathit{false}\}$ in the first case, $\mathrm{Pht}_\varphi = \{\mathit{true}, \mathit{false}\}^3$ in the second case, $\mathrm{Pht}_\varphi = \mathrm{Pht}_{\psi_1} \times \mathrm{Pht}_{\psi_2}$ in the third case, $\mathrm{Pht}_\varphi = \mathrm{Pht}_\psi$ in the fourth case, and $\mathrm{Pht}_\varphi = \mathcal{P}(\mathrm{Pht}_\psi)$ in the last case.

We are particularly interested in φ-phenotypes for valuations that map all set variables to the empty set. Thus for a tree t, by $[t]_\varphi$ we denote $[t \otimes v_\emptyset]_\varphi$, where $v_\emptyset$ is the valuation mapping all free variables of φ to the empty set.

We immediately see two facts. First, Pht_φ is finite for every φ. Second, the fact whether φ holds in $\hat{t}$ is determined by $[\hat{t}]_\varphi$.

Next, we observe that phenotypes of trees behave in a compositional way, as formalized below. For every symbol a and every formula φ, define a function $\mathrm{Comp}_{a,\varphi} \colon \mathcal{P}(\mathcal{V}) \times (\mathrm{Pht}_\varphi)^r \to \mathrm{Pht}_\varphi$, where r is the rank of a, and $\mathcal{V}$ is the set of variables that may occur in formulae of Weak MSO. The functions are defined so that the following lemma holds.

▶ **Lemma 3.1.** *Let φ be a formula with free variables contained in $\mathcal{F}$, and let $\hat{t} = t \otimes v$ be an $\mathcal{F}$-tree with root labeled by a symbol a of rank r. If $\mathcal{R}$ is the set those variables $X \in \mathcal{F}$ for which $v(X)$ contains the root of t, and $\hat{t}_i$ is the subtree of $\hat{t}$ rooted at the i-th child of the root (for $i \in \{1, \dots, r\}$), then $[\hat{t}]_\varphi = \mathit{Comp}_{a,\varphi}(\mathcal{R}, [\hat{t}_1]_\varphi, \dots, [\hat{t}_r]_\varphi)$.*

While defining $\mathrm{Comp}_{a,\varphi}$ we proceed by induction on the size of φ.

When φ is of the form $b(X)$ or $X \subseteq Y$, then we see that φ holds in $\hat{t}$ iff it holds in every subtree $\hat{t}_i$ and in the root of $\hat{t}$. Thus for $\varphi = b(X)$ as $\mathrm{Comp}_{a,\varphi}(\mathcal{R}, \tau_1, \dots, \tau_r)$ we take *true* when $\tau_i = \mathit{true}$ for all $i \in \{1, \dots, r\}$ and either $a = b$ or $X \notin \mathcal{R}$. For $\varphi = (X \subseteq Y)$ the last part of the condition is replaced by „if $X \in \mathcal{R}$ then $Y \in \mathcal{R}$".

Next, suppose that $\varphi = (X <_i Y)$. There are several ways in which X and Y can be distributed between the subtrees $\hat{t}_j$ and the root of $\hat{t}$ so that $X <_i Y$ holds in $\hat{t}$ (i.e. so that the first coordinate of $[t]_\varphi$ is *true*), but in any case, to determine whether $X <_i Y$ holds in $\hat{t}$ it is enough to know whether X contains the root of $\hat{t}$, whether Y contains the root of $\hat{t}$, and for each $j \in \{1, \dots, r\}$ whether X is nonempty in $\hat{t}_j$, whether Y is nonempty in $\hat{t}_j$, and whether $X <_i Y$ holds in $\hat{t}_j$. These facts are determined by the set $\mathcal{R}$ and by the φ-phenotypes of $\hat{t}_j$. For the last two parts of the φ-phenotype the situation is even more direct, as X is empty in $\hat{t}$ iff it does not contain the root of $\hat{t}$ and is empty in all $\hat{t}_j$. The function $\mathrm{Comp}_{a,\varphi}$ is defined appropriately.

When $\varphi = (\psi_1 \wedge \psi_2)$ as $\mathrm{Comp}_{a,\varphi}(\mathcal{R}, \tau_1, \dots, \tau_r)$ we take the pair of $\mathrm{Comp}_{a,\psi_i}(\mathcal{R}, \tau_1, \dots, \tau_r)$ for $i \in \{1, 2\}$, and when $\varphi = (\neg\psi)$, we simply take $\mathrm{Comp}_{a,\varphi} = \mathrm{Comp}_{a,\psi}$.

Finally, suppose that $\varphi = \exists_{\mathsf{fin}} X.\psi$. In this situation we see that the proper definition of $\mathrm{Comp}_{a,\varphi}(\mathcal{R}, \tau_1, \dots, \tau_r)$ is

$$\{\mathrm{Comp}_{a,\psi}(\mathcal{R} \setminus \{X\}, \sigma_1, \dots, \sigma_r), \mathrm{Comp}_{a,\psi}(\mathcal{R} \cup \{X\}, \sigma_1, \dots, \sigma_r) \mid \sigma_1 \in \tau_1, \dots, \sigma_r \in \tau_r\}\,.$$

4 Values of Terms

In this section, we introduce the models announced in Theorem 1.1. At the end, we need the models only for sentences, but, in order to perform induction, we define them also for formulae with free variables.

Fix a formula φ of Weak MSO. We will construct a finitary model $\mathcal{M}_\varphi$ of infinitary λ-calculus, i.e., define a finite set $D_{\mathcal{M}_\varphi}[\alpha]$ for each type α, and its element $[\![K]\!]^v_{\mathcal{M}_\varphi}$ for each closed term K of type α and each $D_{\mathcal{M}_\varphi}$-valuation v with $\mathrm{dom}(v) \supseteq FV(K)$. Instead of $D_{\mathcal{M}_\varphi}[\alpha]$ and $[\![K]\!]^v_{\mathcal{M}_\varphi}$ we simply write $D_\varphi[\alpha]$ and $[\![K]\!]^v_\varphi$. When K is closed, we omit the superscript v, and we call $[\![K]\!]_\varphi$ the *φ-value* of K. Additionally, we will define a function $\mathrm{pht}_\varphi : D_\varphi[o] \to \mathrm{Pht}_\varphi$, with the intention that $\mathrm{pht}_\varphi([\![K^o]\!]_\varphi) = [BT(K)]_\varphi$.

The model is defined by induction on the formula φ. When $x = (x_1, \dots, x_k)$ is a tuple, we write $\pi_i(x)$ for x_i.

If φ is an atomic formula, then $D_\varphi[\alpha] = \{\top\}$ and $[\![K]\!]^v_\varphi = \top$. The function pht_φ maps $\top \in \mathbb{D}_\varphi[o]$ to the unique φ-phenotype that is of the form $[t]_\varphi$ (when all set variables are mapped to the empty set, the φ-phenotype does not depend on the tree t).

If $\varphi = (\psi_1 \wedge \psi_2)$, then we take $D_\varphi[\alpha] = D_{\psi_1}[\alpha] \times D_{\psi_2}[\alpha]$, and $[\![K]\!]^v_\varphi = ([\![K]\!]^{\pi_1 \circ v}_{\psi_1}, [\![K]\!]^{\pi_2 \circ v}_{\psi_2})$, and $\mathrm{pht}_\varphi((\tau_1, \tau_2)) = (\mathrm{pht}_{\psi_1}(\tau_1), \mathrm{pht}_{\psi_2}(\tau_2))$. If $\varphi = (\neg\psi)$, then simply $D_\varphi[\alpha] = D_\psi[\alpha]$, and $[\![K]\!]^v_\varphi = [\![K]\!]^v_\psi$, and $\mathrm{pht}_\varphi = \mathrm{pht}_\psi$.

Suppose now that $\varphi = \exists_{\mathsf{fin}} X.\psi$; this case is slightly more complicated. We start by defining the domain $D_\varphi[\alpha]$, by induction on the type α, as follows:

$$\begin{aligned} &D_\varphi[\alpha] = D^1_\varphi[\alpha] \times D_\psi[\alpha]\,, \qquad \text{where} \\ &D^1_\varphi[\alpha_1 \to \dots \to \alpha_n \to o] = (\mathrm{Pht}_\varphi)^{D_\varphi[\alpha_1] \times \dots \times D_\varphi[\alpha_n]}\,. \end{aligned}$$

For $\sigma \in D_\varphi[\alpha \to \beta]$ and $\tau \in D_\varphi[\alpha]$, the composition $(\sigma\,\tau) \in D_\varphi[\beta]$ is defined by

$$\begin{aligned} &\pi_1(\sigma\,\tau)(\rho_1, \dots, \rho_n) = \pi_1(\sigma)(\tau, \rho_1, \dots, \rho_n) \qquad \text{for all arguments } \rho_1, \dots, \rho_n, \text{ and} \\ &\pi_2(\sigma\,\tau) = (\pi_2(\sigma)\,\pi_2(\tau))\,. \end{aligned}$$

Next, for each term K and each φ-valuation v with $\mathrm{dom}(v) \supseteq FV(K)$, the value $[\![K]\!]^v_\varphi$ is defined using least fixpoints, as follows. A *φ-evaluation* is a function that maps every term K^α and every φ-valuation v with $\mathrm{dom}(v) \supseteq FV(K)$ to an element of $D^1_\varphi[\alpha]$. Recall that Pht_φ is a set, so the inclusion order on this set induces an order on φ-evaluations that is a complete lattice. Namely, for $\hat\sigma, \hat\tau \in D^1_\varphi[\alpha_1 \to \dots \to \alpha_n \to o]$ we say that $\hat\sigma \subseteq \hat\tau$ if for all $\rho_1 \in D_\varphi[\alpha_1], \dots, \rho_n \in D_\varphi[\alpha_n]$ it holds that $\hat\sigma(\rho_1, \dots, \rho_n) \subseteq \hat\tau(\rho_1, \dots, \rho_n)$, and for two φ-evaluations p, q we say that $p \subseteq q$ if for every term K and every φ-valuation v with $\mathrm{dom}(v) \supseteq FV(K)$ it holds that $p(K, v) \subseteq q(K, v)$. We define an operation $\mathcal{F}_\varphi$ on φ-evaluations. Let K, v and $\rho_1, \dots, \rho_n$ be as above. We take

$$\mathcal{F}_\varphi(p)(K, v)(\rho_1, \dots, \rho_n) = \{\mathrm{pht}_\psi([\![K]\!]^{\pi_2 \circ v}_\psi\, \pi_2(\rho_1) \dots \pi_2(\rho_n))\} \cup \eta\,, \qquad \text{where}$$

$$\eta = \begin{cases} p(M, v)((p(N, v), [\![N]\!]^{\pi_2 \circ v}_\psi), \rho_1, \dots, \rho_n) & \text{if } K = M\,N, \\ p(M, v[x \mapsto \rho_1])(\rho_2, \dots, \rho_n) & \text{if } K = \lambda x.M, \\ \pi_1(v(x))(\rho_1, \dots, \rho_n) & \text{if } K = x, \\ \mathrm{Comp}_{a,\varphi}(\emptyset, \pi_1(\rho_1), \dots, \pi_1(\rho_n)) & \text{if } K = a. \end{cases}$$

It is clear that the operation $\mathcal{F}_\varphi$ is monotone, and hence we can consider its least fixpoint. We take $[\![K]\!]^v_\varphi = (LFP(\mathcal{F}_\varphi)(K, v), [\![K]\!]^{\pi_2 \circ v}_\psi)$, and $\mathrm{pht}_\varphi = \pi_1$.

The first component in the definition of $\mathcal{F}_\varphi(p)$, i.e. $\mathrm{pht}_\psi([\![K]\!]^{\pi_2 \circ v}_\psi\, \pi_2(\rho_1) \dots \pi_2(\rho_n))$, intuitively corresponds to the possibility that the set X, over which we quantify, is empty (and then we can simply read the phenotype for the subformula ψ). In the second component, η, we look inside the term, and, basically, we compose the results from its subterms. In particular, when K is a symbol ($K = a$), the call to $\mathrm{Comp}_{a,\varphi}$ takes into account both the possibility that X contains the a-labeled root, and that it does not contain it. Since we take the least fixpoint, we descend to subterms only finitely many times, and after that we have to take the first component (or end with $K = x$ or $K = a$); this corresponds to the fact that the quantification ranges over finite sets only.

We need to observe that our construction indeed satisfies conditions M1–M4 of a model. Condition M1 is clear. The other conditions are established in the following lemmata.

▶ **Lemma 4.1.** *For every term of the form $M\,N$, every formula φ of* Weak MSO*, and every φ-valuation v with $dom(v) \supseteq FV(M\,N)$ it holds that $[\![M\,N]\!]^v_\varphi = ([\![M]\!]^v_\varphi\,[\![N]\!]^v_\varphi)$.*

Proof. The proof is by induction on φ. The lemma is obvious for atomic φ, and for conjunctions and negations it follows immediately from the induction assumption. Suppose thus that $\varphi = \exists_{\mathsf{fin}} X.\psi$. The second coordinates of $[\![M\,N]\!]^v_\varphi$ and $[\![M]\!]^v_\varphi\,[\![N]\!]^v_\varphi$ contain ψ-values, equal by the induction assumption. It remains to prove for all arguments $\rho_1, \dots, \rho_n$ that

$$\pi_1([\![M\,N]\!]^v_\varphi)(\rho_1, \dots, \rho_n) = \pi_1([\![M]\!]^v_\varphi)([\![N]\!]^v_\varphi, \rho_1, \dots, \rho_n)\,. \tag{1}$$

Because $\pi_1([\![\cdot]\!]^{\cdot}_\varphi)$ is a fixpoint of the $\mathcal{F}_\varphi$ operation, by the definition of $\mathcal{F}_\varphi$ we have

$$\pi_1([\![M\,N]\!]^v_\varphi)(\rho_1, \dots, \rho_n) = \{\xi\} \cup \pi_1([\![M]\!]^v_\varphi)([\![N]\!]^v_\varphi, \rho_1, \dots, \rho_n)\,, \qquad \text{where}$$
$$\xi = \mathrm{pht}_\psi([\![M\,N]\!]^{\pi_2\circ v}_\psi\ \pi_2(\rho_1)\ \dots\ \pi_2(\rho_n))\,.$$

By the induction assumption we have $[\![M\,N]\!]^{\pi_2\circ v}_\psi = [\![M]\!]^{\pi_2\circ v}_\psi\,[\![N]\!]^{\pi_2\circ v}_\psi = [\![M]\!]^{\pi_2\circ v}_\psi\,\pi_2([\![N]\!]^v_\varphi)$, Once again looking at the definition of $\mathcal{F}_\varphi$ (this time for the term M) we conclude that ξ belongs to the set $\pi_1([\![M]\!]^v_\varphi)([\![N]\!]^v_\varphi, \rho_1, \dots, \rho_n)$, which yields (1). ◀

▶ **Lemma 4.2.** *For every term of the form $\lambda x^\alpha.M$, every formula φ of* Weak MSO*, every φ-valuation v with $dom(v) \supseteq FV(\lambda x.M)$, and every $\tau \in D_\varphi[\alpha]$ it holds that $([\![\lambda x.M]\!]^v_\varphi\,\tau) = [\![M]\!]^{v[x\mapsto\tau]}_\varphi$.*

Proof. Very similar to the previous one. ◀

▶ **Lemma 4.3.** *For every term of the form $K[M/x]$, every formula φ of* Weak MSO*, and every φ-valuation v with $dom(v) \supseteq FV(K[M/x]) \cup FV(M)$ it holds that $[\![K[M/x]]\!]^v_\varphi = [\![K]\!]^{v[x\mapsto[\![M]\!]^v_\varphi]}_\varphi$.*

Proof. The proof bases on the fact that $[\![x[M/x]]\!]^v_\varphi = [\![M]\!]^v_\varphi = [\![x]\!]^{v[x\mapsto[\![M]\!]^v_\varphi]}_\varphi$. Since $K[M/x]$ is obtained by replacing in K every x by M, and the φ-phenotypes are defined in a compositional way, using standard techniques it is easy to conclude that $[\![K[M/x]]\!]^v_\psi = [\![K]\!]^{v[x\mapsto[\![M]\!]^v_\varphi]}_\varphi$. ◀

We now come to the crucial lemma saying that the model actually computes φ-phenotypes of generated trees.

▶ **Lemma 4.4.** *For every closed term P of type o, and every formula φ of* Weak MSO *it holds that $pht_\varphi([\![P]\!]_\varphi) = [BT(P)]_\varphi$.*

Proof. The proof is by induction on φ. The lemma is obvious when φ is atomic, and it immediately follows from the induction assumption when φ is a conjunction or a negation. In the sequel we assume that $\varphi = \exists_{\mathsf{fin}} X.\psi$. In this case a φ-phenotype is a set, and thus we have to prove two inclusions. Recall that $\mathrm{pht}_\varphi = \pi_1$.

We first concentrate on the inclusion $\pi_1(\llbracket P\rrbracket_\varphi) \subseteq [BT(P)]_\varphi$ (soundness). In fact, we prove a generalized statement, talking about terms of all types, and not necessarily closed; to state it, we need to say what it means for a φ-value to be sound for a term. We define it by induction on the type. Let K be a closed term of type $\alpha = \alpha_1 \to \cdots \to \alpha_n \to o$; assume that we already know which φ-values are sound for closed terms of types $\alpha_1, \dots, \alpha_n$. We say that $\hat{\sigma} \in D^1_\varphi[\alpha]$ is *sound* for K, if whenever some $\rho_1 \in D_\varphi[\alpha_1], \dots, \rho_n \in D_\varphi[\alpha_n]$ are sound for some closed terms $Q_1, \dots, Q_n$, respectively, then $\hat{\sigma}(\rho_1, \dots, \rho_n) \subseteq [BT(K\,Q_1\,\dots\,Q_n)]_\varphi$. We say that $\sigma \in D_\varphi[\alpha]$ is *sound* for K if $\pi_1(\sigma)$ is sound for K, and $\pi_2(\sigma) = \llbracket K\rrbracket_\psi$.

Next, we define soundness for terms with free variables. We say that a φ-valuation v is *sound* for a substitution (i.e. a partial mapping from variables to closed terms) θ if $v(x)$ is sound for $\theta(x)$ for every $x \in \mathrm{dom}(v) = \mathrm{dom}(\theta)$. We say that $\sigma \in D_\varphi[\alpha]$ is *v-sound* for a term K^α, where $FV(K) \subseteq \mathrm{dom}(v)$, if whenever v is sound for a substitution θ then σ is sound for $K\theta$. When p is a φ-evaluation, we say that it is *sound* if $p(K,v)$ is v-sound for K, for all arguments K, v.

Recall that $\pi_1(\llbracket\cdot\rrbracket^{\cdot}_\varphi)$ is the least fixpoint of the $\mathcal{F}_\varphi$ operation. We want to prove that it is sound. In order to prove some property (e.g. soundness) for the least fixpoint, it is enough to prove three things: that the property holds for the minimal element of the lattice, that it is preserved by the operation, and that it is preserved by taking the least upper bound. In our case the minimal φ-evaluation p_0 maps all arguments to the empty set, so it is clearly sound, since the empty set is a subset of every set. Consider now some set $\mathcal{P}$ of sound φ-evaluations. Its least upper bound $\bigcup\mathcal{P}$ is defined by $(\bigcup\mathcal{P})(K,v)(\rho_1,\dots,\rho_n) = \bigcup_{p\in\mathcal{P}} p(K,v)(\rho_1,\dots,\rho_n)$ for all arguments $K, v, \rho_1, \dots, \rho_n$. It is thus clear that $\bigcup\mathcal{P}$ is sound, since whenever $\xi \in (\bigcup\mathcal{P})(K,v)(\rho_1,\dots,\rho_n)$, then $\xi \in p(K)(\rho_1,\dots,\rho_n)$ for some $p \in \mathcal{P}$. In order to obtain that the least fixpoint $\pi_1(\llbracket\cdot\rrbracket^{\cdot}_\varphi)$ is sound, it remains to prove the following claim.

▶ Claim. If p is a sound φ-evaluation, then $\mathcal{F}_\varphi(p)$ is sound as well.

Proof. The proof is by a straightforward analysis of definitions of $\mathcal{F}_\varphi$ and of soundness. The details follow.

Let K be a term of type $\alpha = \alpha_1 \to \cdots \to \alpha_n \to o$, let v be a φ-valuation that is sound for a substitution θ, where $FV(K) \subseteq \mathrm{dom}(v)$, let $\rho_1 \in D_\varphi[\alpha_1], \dots, \rho_n \in D_\varphi[\alpha_n]$ be sound for some closed terms $Q_1, \dots, Q_n$ (respectively), and let $\xi \in \mathcal{F}_\varphi(p,v)(\rho_1,\dots,\rho_n)$. Denote $t = BT(K\theta\,Q_1\,\dots\,Q_n)$. We need to prove that $\xi \in [t]_\varphi$.

We follow the definition of $\mathcal{F}_\varphi(p)$. It is possible that $\xi = \mathrm{pht}_\psi(\llbracket K\rrbracket^{\pi_2\circ v}_\psi\,\pi_2(\rho_1)\,\dots\,\pi_2(\rho_n))$. Then, by the definition of soundness we have that $\pi_2(\rho_i) = \llbracket Q_i\rrbracket_\psi$ for all $i \in \{1,\dots,n\}$, and that $\pi_2(v(x)) = \llbracket\theta(x)\rrbracket_\psi$ for all $x \in FV(K)$; thus by Lemmata 4.3 and 4.1 we have $\xi = \mathrm{pht}_\psi(\llbracket K\theta\,Q_1\,\dots\,Q_n\rrbracket_\psi)$. The induction assumption of the whole lemma implies that $\xi = [t]_\psi$, and thus $\xi \in [t]_\varphi$ by the definition of a φ-phenotype of a tree.

Next, suppose that $K = M\,N$ and $\xi \in \eta = p(M,v)((p(N,v), \llbracket N\rrbracket^{\pi_2\circ v}_\psi), \rho_1, \dots, \rho_n)$. By assumption $p(M,v)$ is v-sound for M, hence sound for $M\theta$, and $p(N,v)$ is v-sound for N, hence sound for $N\theta$. Because $\pi_2(v(x)) = \llbracket\theta(x)\rrbracket_\psi$ for all $x \in FV(K)$, by Lemma 4.3 we have $\llbracket N\rrbracket^{\pi_2\circ v}_\psi = \llbracket N\theta\rrbracket_\psi$, so $(p(N,v), \llbracket N\rrbracket^{\pi_2\circ v}_\psi)$ is sound for $N\theta$. Noticing that $M\theta\,(N\theta) = K\theta$, and recalling that ρ_i is is sound for Q_i, for every i, it follows from the definition of soundness that $\eta \subseteq [t]_\varphi$.

Another possibility is that $K = \lambda x.M$ and $\xi \in \eta = p(M, v[x \mapsto \rho_1])(\rho_2,\dots,\rho_n)$. Notice that $v' = v[x \mapsto \rho_1]$ is sound for $\theta' = \theta[x \mapsto Q_1]$. By assumption $p(M,v')$ is v'-sound for M, so sound for $M\theta'$. This implies that η is a subset of the φ-phenotype of the tree generated by $M\theta'\,Q_2\,\dots\,Q_n$, hence of t (since $M\theta'$ is β-equivalent to $K\,Q_1$).

Yet another possibility is that $K = x$ and $\xi \in \eta = \pi_1(v(x))(\rho_1,\dots,\rho_n)$. Then, by soundness of ρ_i and $v(x)$, we obtain that η is a subset of the φ-phenotype of $BT(\theta(x)\,Q_1\,\dots\,Q_n) = t$.

Finally, it is possible that $K = a$ and $\xi \in \eta = \mathrm{Comp}_{a,\varphi}(\emptyset, \pi_1(\rho_1), \dots, \pi_1(\rho_n))$. Notice that the types $\alpha_1, \dots, \alpha_n$ are o, and that n is the rank of a. By soundness of ρ_i we have that $\pi_1(\rho_i) \subseteq [BT(Q_i)]_\varphi$ for $i \in \{1, \dots, n\}$. The root of t has label a, and the subtrees starting in its children are $BT(Q_i)$, so by monotonicity of $\mathrm{Comp}_{a,\varphi}$ we obtain

$$\xi \in \mathrm{Comp}_{a,\varphi}(\emptyset, \pi_1(\rho_1), \dots, \pi_1(\rho_n)) \subseteq \mathrm{Comp}_{a,\varphi}(\emptyset, [BT(Q_1)]_\varphi, \dots, [BT(Q_n)]_\varphi) = [t]_\varphi \,. \quad ◀$$

This finishes the proof of the „soundness" inclusion $\pi_1(\llbracket P \rrbracket_\varphi) \subseteq [BT(P)]_\varphi$. We now turn into the „completeness" inclusion, i.e. $[BT(P)]_\varphi \subseteq \pi_1(\llbracket P \rrbracket_\varphi)$.

Let X be a finite set of nodes of the tree generated by a term K. The following definition is by induction on the depth (distance from the root) of the deepest node in X. We say that K is *expanded up to* X if either

- X is empty, or
- $K = a\, K_1 \, \dots \, K_r$, and K_i is expanded up to the set $X{\restriction}_{BT(K_i)}$, for all $i \in \{1, \dots, r\}$.

In other words, it is required that all nodes from X and their ancestors can be generated from K without performing any β-reductions. We prove the following claim.

▶ Claim. Let K be a closed term of type o that is expanded up to a finite set X, and let v be the valuation that maps the variable X to the set X, and all free variables of φ to the empty set. Then $[BT(K) \otimes v]_\psi \in \pi_1(\llbracket K \rrbracket_\varphi)$.

Proof. The proof is by induction on the depth (distance from the root) of the deepest node in X. We have two cases. Suppose first that X is empty. Then $[BT(K) \otimes v]_\psi = [BT(K)]_\psi = \mathrm{pht}_\psi(\llbracket K \rrbracket_\psi)$ by the induction assumption of the whole lemma. By definition of the $\mathcal{F}_\varphi$ operation, we see that $\mathrm{pht}_\psi(\llbracket K \rrbracket_\psi)$ is an element of $\pi_1(\llbracket K \rrbracket_\varphi)$, as required.

Another possibility is that $K = a\, K_1 \, \dots \, K_r$, and K_i is expanded up to the set $X{\restriction}_{BT(K_i)}$, for all $i \in \{1, \dots, r\}$. Let v_i be the valuation that maps variable X to the set $X{\restriction}_{BT(K_i)}$, and all free variables of φ to the empty set. Lemma 3.1 says that $[BT(K) \otimes v]_\psi = \mathrm{Comp}_{a,\psi}(\mathcal{R}, [BT(K_1) \otimes v_1]_\psi, \dots, [BT(K_r) \otimes v_r]_\psi)$, where $\mathcal{R} = \{X\}$ if X contains the root of $BT(K)$, and $\mathcal{R} = \emptyset$ otherwise. Denote $\hat{\rho}_i = \pi_1(\llbracket K_i \rrbracket_\varphi)$, for all i. The induction assumption implies that $[BT(K_i) \otimes v_i]_\psi \in \hat{\rho}_i$, so by definition of $\mathrm{Comp}_{a,\varphi}$ we have $[BT(K) \otimes v]_\psi \in \mathrm{Comp}_{a,\varphi}(\emptyset, \hat{\rho}_1, \dots, \hat{\rho}_r)$. Finally, by the definition of $\mathcal{F}_\varphi$ and by Lemma 4.1 we obtain

$$[BT(K) \otimes v]_\psi \in \mathrm{Comp}_{a,\varphi}(\emptyset, \hat{\rho}_1, \dots, \hat{\rho}_r) \subseteq \pi_1(\llbracket a \rrbracket_\varphi)(\llbracket K_1 \rrbracket_\varphi, \dots, \llbracket K_r \rrbracket_\varphi) = \pi_1(\llbracket K \rrbracket_\varphi) \,. \quad ◀$$

We come back to the proof of the inclusion $[BT(P)]_\varphi \subseteq \pi_1(\llbracket P \rrbracket_\varphi)$. Take some $\xi \in [BT(P)]_\varphi$. By the definition of the φ-phenotype, $\xi = [BT(P) \otimes v]_\psi$ for some valuation v that maps the variable X to some finite set X of Σ-labeled nodes of $BT(P)$, and all free variables of φ to the empty set. It should be clear that after performing appropriately many head β-reductions from P one obtains a term P' that is expanded up to the set X (thanks to the fact that X does not contain $\perp$-labeled nodes). By the above claim we have that $\xi \in \pi_1(\llbracket P' \rrbracket_\varphi)$, and Fact 2.1 implies that $\llbracket P' \rrbracket_\varphi = \llbracket P \rrbracket_\varphi$. This finishes the proof of the inclusion $[BT(P)]_\varphi \subseteq \pi_1(\llbracket P \rrbracket_\varphi)$. ◀

4.1 Effectiveness

Thanks to Lemma 4.4 we obtain, for every sentence φ of Weak MSO, that our model recognizes the set of trees in which φ holds. In order to obtain Theorem 1.1, we need to observe that this model is effective, i.e., that the φ-value of a λY-term K can be computed from K and from φ.

This boils down to the question how to compute $[\![K]\!]_\varphi$ in the situation when $\varphi = \exists_{\mathsf{fin}} X.\psi$. Then the definition uses the least fixpoint of the $\mathcal{F}_\varphi$ operation on φ-evaluations. Although a φ-evaluation is an infinite object, we do not need to compute it for all arguments. Namely, a pair of arguments (M, v) is called *interesting*, if M is a subterm of K, and $\mathrm{dom}(v)$ contains only variables that appear in K. When (M, v) is interesting, we notice that in order to compute $\mathcal{F}_\varphi(p)(M, v)$ using the definition of $\mathcal{F}_\varphi$, it is enough to know what p returns for interesting pairs. Moreover, there are only finitely many interesting pairs: the λY-term K has only finitely many different subterms (even while being seen as an infinitary λ-term). It follows that there are only finitely many ways how a φ-evaluation may behave on interesting arguments, and thus the least fixpoint of $\mathcal{F}_\varphi$ on these arguments can be computed by repeatedly applying $\mathcal{F}_\varphi$ to the minimal φ-evaluation.

5 Conclusions

We have shown how, given a sentence of Weak MSO, one can construct an effective and finitary model of simply typed λ-calculus, that recognizes the set of trees in which the sentence is true. Whereas this result was obtained earlier in [20] by employing parity automata corresponding to Weak MSO, we define the model directly from the formula. We do not claim that our method is better or simpler, it is just different. We believe that the logical approach may be potentially useful when considering other weak logics, for which there is no natural automata model. In fact, it is an ongoing work to prove that the same result holds for the WMSO+U logic, which extends Weak MSO by a quantifier U, expressing that a subformula holds for arbitrarily large finite sets [3].

Acknowledgments. We are very grateful to Sylvain Salvati for his patience and illuminating discussions, and to reviewers of the previous version of this paper for their useful comments.

References

1 Klaus Aehlig. A finite semantics of simply-typed lambda terms for infinite runs of automata. *Logical Methods in Computer Science*, 3(3), 2007. `doi:10.2168/LMCS-3(3:1)2007`.

2 H. P. Barendregt. *The Lambda Calculus Its Syntax and Semantics*, volume 103. North Holland, revised edition, 1984.

3 Mikolaj Bojańczyk and Szymon Toruńczyk. Weak MSO+U over infinite trees. In Christoph Dürr and Thomas Wilke, editors, *29th International Symposium on Theoretical Aspects of Computer Science, STACS 2012, February 29th – March 3rd, 2012, Paris, France*, volume 14 of *LIPIcs*, pages 648–660. Schloss Dagstuhl – Leibniz-Zentrum fuer Informatik, 2012. `doi:10.4230/LIPIcs.STACS.2012.648`.

4 Christopher H. Broadbent, Arnaud Carayol, C.-H. Luke Ong, and Olivier Serre. Recursion schemes and logical reflection. In *Proceedings of the 25th Annual IEEE Symposium on Logic in Computer Science, LICS 2010, 11-14 July 2010, Edinburgh, United Kingdom*, pages 120–129. IEEE Computer Society, 2010. `doi:10.1109/LICS.2010.40`.

5 Lorenzo Clemente, Paweł Parys, Sylvain Salvati, and Igor Walukiewicz. Ordered tree-pushdown systems. In Prahladh Harsha and G. Ramalingam, editors, *35th IARCS Annual Conference on Foundation of Software Technology and Theoretical Computer Science, FSTTCS 2015, December 16-18, 2015, Bangalore, India*, volume 45 of *LIPIcs*, pages 163–177. Schloss Dagstuhl – Leibniz-Zentrum fuer Informatik, 2015. URL: `http://www.dagstuhl.de/dagpub/978-3-939897-97-2`, `doi:10.4230/LIPIcs.FSTTCS.2015.163`.

6 Werner Damm. The IO- and OI-hierarchies. *Theoretical Computer Science*, 20(2):95–207, 1982. `doi:10.1016/0304-3975(82)90009-3`.

7 Charles Grellois and Paul-André Melliès. Finitary semantics of linear logic and higher-order model-checking. In Giuseppe F. Italiano, Giovanni Pighizzini, and Donald Sannella, editors, *Mathematical Foundations of Computer Science 2015 – 40th International Symposium, MFCS 2015, Milan, Italy, August 24-28, 2015, Proceedings, Part I*, volume 9234 of *Lecture Notes in Computer Science*, pages 256–268. Springer, 2015. `doi:10.1007/978-3-662-48057-1_20`.

8 Axel Haddad. Model checking and functional program transformations. In Seth and Vishnoi [22], pages 115–126. `doi:10.4230/LIPIcs.FSTTCS.2013.115`.

9 Matthew Hague, Andrzej S. Murawski, C.-H. Luke Ong, and Olivier Serre. Collapsible pushdown automata and recursion schemes. In *Proceedings of the Twenty-Third Annual IEEE Symposium on Logic in Computer Science, LICS 2008, 24-27 June 2008, Pittsburgh, PA, USA*, pages 452–461. IEEE Computer Society, 2008. `doi:10.1109/LICS.2008.34`.

10 Thomas A. Henzinger and Dale Miller, editors. *Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS), CSL-LICS'14, Vienna, Austria, July 14-18, 2014*. ACM, 2014.

11 Martin Hofmann and Wei Chen. Abstract interpretation from Büchi automata. In Henzinger and Miller [10], pages 51:1–51:10. `doi:10.1145/2603088.2603127`.

12 Teodor Knapik, Damian Niwiński, and Paweł Urzyczyn. Higher-order pushdown trees are easy. In Mogens Nielsen and Uffe Engberg, editors, *Foundations of Software Science and Computation Structures, 5th International Conference, FOSSACS 2002. Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2002 Grenoble, France, April 8-12, 2002, Proceedings*, volume 2303 of *Lecture Notes in Computer Science*, pages 205–222. Springer, 2002. `doi:10.1007/3-540-45931-6_15`.

13 Naoki Kobayashi and C.-H. Luke Ong. A type system equivalent to the modal mu-calculus model checking of higher-order recursion schemes. In *Proceedings of the 24th Annual IEEE Symposium on Logic in Computer Science, LICS 2009, 11-14 August 2009, Los Angeles, CA, USA*, pages 179–188. IEEE Computer Society, 2009. `doi:10.1109/LICS.2009.29`.

14 Giulio Manzonetto. Models and theories of lambda calculus. *CoRR*, abs/0904.4756, 2009. URL: `http://arxiv.org/abs/0904.4756`.

15 Albert R. Meyer. What is a model of the lambda calculus? *Information and Control*, 52(1):87–122, 1982. `doi:10.1016/S0019-9958(82)80087-9`.

16 C.-H. Luke Ong. On model-checking trees generated by higher-order recursion schemes. In *21th IEEE Symposium on Logic in Computer Science (LICS 2006), 12-15 August 2006, Seattle, WA, USA, Proceedings*, pages 81–90. IEEE Computer Society, 2006. `doi:10.1109/LICS.2006.38`.

17 Sylvain Salvati and Igor Walukiewicz. Evaluation is MSOL-compatible. In Seth and Vishnoi [22], pages 103–114. `doi:10.4230/LIPIcs.FSTTCS.2013.103`.

18 Sylvain Salvati and Igor Walukiewicz. Krivine machines and higher-order schemes. *Inf. Comput.*, 239:340–355, 2014. `doi:10.1016/j.ic.2014.07.012`.

19 Sylvain Salvati and Igor Walukiewicz. A model for behavioural properties of higher-order programs. In Stephan Kreutzer, editor, *24th EACSL Annual Conference on Computer Science Logic, CSL 2015, September 7-10, 2015, Berlin, Germany*, volume 41 of *LIPIcs*, pages 229–243. Schloss Dagstuhl – Leibniz-Zentrum fuer Informatik, 2015. URL: `http://www.dagstuhl.de/dagpub/978-3-939897-90-3`, `doi:10.4230/LIPIcs.CSL.2015.229`.

20 Sylvain Salvati and Igor Walukiewicz. Typing weak MSOL properties. In Andrew M. Pitts, editor, *Foundations of Software Science and Computation Structures – 18th International Conference, FoSSaCS 2015, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2015, London, UK, April 11-18, 2015. Proceedings*, volume

9034 of *Lecture Notes in Computer Science*, pages 343–357. Springer, 2015. doi:10.1007/978-3-662-46678-0_22.

21 Sylvain Salvati and Igor Walukiewicz. Using models to model-check recursive schemes. *Logical Methods in Computer Science*, 11(2), 2015. doi:10.2168/LMCS-11(2:7)2015.

22 Anil Seth and Nisheeth K. Vishnoi, editors. *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2013, December 12-14, 2013, Guwahati, India*, volume 24 of *LIPIcs*. Schloss Dagstuhl – Leibniz-Zentrum fuer Informatik, 2013.

23 Takeshi Tsukada and C.-H. Luke Ong. Compositional higher-order model checking via ω-regular games over Böhm trees. In Henzinger and Miller [10], pages 78:1–78:10. doi:10.1145/2603088.2603133.