A Finite Semantics of Simply-Typed Lambda Terms for Infinite Runs of Automata

Klaus Aehlig

Mathematisches Institut
Ludwig-Maximilians-Universität München
Theresienstr. 39, 80333 München, Germany
aehlig@math.lmu.de

Abstract. Model checking properties are often described by means of finite automata. Any particular such automaton divides the set of infinite trees into finitely many classes, according to which state has an infinite run. Building the full type hierarchy upon this interpretation of the base type gives a finite semantics for simply-typed lambda-trees.
A calculus based on this semantics is proven sound and complete. In particular, for regular infinite lambda-trees it is decidable whether a given automaton has a run or not. As regular lambda-trees are precisely recursion schemes, this decidability result holds for arbitrary recursion schemes of arbitrary level, without any syntactical restriction. This partially solves an open problem of Knapik, Niwinski and Urzyczyn.

1 Introduction and Related Work

The lambda calculus has long been used as a model of computation. Restricting it to simple types allows for a particularly simple set-theoretic semantics. The drawback, however, is that only few functions can be defined in the simply-typed lambda calculus. To overcome this problem one can, for example, add fixed-point combinators Y_σ at every type, or allow infinitary lambda terms. The latter is more flexible, as we can always syntactically unfold fixed points, paying the price to obtain an infinite, but regular, lambda-tree.

Finite automata are a standard tool in the realm of model checking [10]. They provide a concrete machine model for the properties to be verified. In this article we combine automata, and hence properties relevant for model checking, with the infinitary simply-typed lambda calculus, using the fact that the standard set theoretic semantics for the simple types has a free parameter — the interpretation of the base type.

More precisely, we consider the following problem.

> Given a, possibly infinite, simply-typed lambda-tree t of base type, and given a non-deterministic tree automaton $\mathfrak{A}$. Does $\mathfrak{A}$ have a run on the normal form of t?

The idea is to provide a "proof" of a run of $\mathfrak{A}$ on the normal form of t by annotating each subterm of t with a semantical value describing how this subterm "looks, as seen by $\mathfrak{A}$". Since, in the end, all the annotations turn out to be out of a fixed finite set, the existence of such a proof is decidable.

So, what does a lambda-tree look like, if seen by an automaton $\mathfrak{A}$? At the base type, a lambda-tree denotes an infinite term. Hence, from $\mathfrak{A}$'s point of view, we have to distinguish for which states there is an infinite run starting in this particular state.

Since we are interested in model checking terms of base type only, we can use any semantics for higher types, as long as it is adequate, that is, sound and complete. So we use the most simple one available, that is, the full set-theoretic semantics with the base type interpreted as just discussed. This yields a finite set as semantical realm for every type.

As an application of the techniques developed in this article, we show that for arbitrary recursion schemes it is decidable whether the defined tree has a property expressible by an automaton with trivial acceptance condition. This gives a partial answer to an open problem by Knapik, Niwinski and Urzyczyn [5].

Infinitary lambda-trees were also considered by Knapik, Niwinski and Urzyczyn [4], who also proved the decidability of the Monadic Second Order (MSO) theory of trees given by recursion schemes enjoying a certain "safety" condition [5]. The fact, that the safety restriction can be dropped at level two has been shown by Aehlig, de Miranda and Ong [2], and, independently, by Knapik, Niwinski, Urzyczyn and Walukiewicz [6]. The work of the former group also uses implicitly the idea of a "proof" that a particular automaton has a run on the normal form of a given infinite lambda-tree.

Recently [9] Luke Ong showed simultaneously and independently that the safety restriction can be dropped for all levels and still decidability for the full MSO theory is obtained. His approach is based on game semantics and is technically quite involved. Therefore, the author believes that his approach, due to its simplicity and straight forwardness, is still of interest, despite showing a weaker result. Moreover, the novel construction of a finite semantics and its adequacy even in a coinductive setting seem to be of independent interest.

2 Preliminaries

Let Σ' be a set of *letters* or *terminals*. We use $\mathfrak{f}$ to denote elements of Σ'. Each terminal $\mathfrak{f}$ is associated an *arity* $\sharp(\mathfrak{f}) \in \mathbb{N}$.

Definition 1. Define $\Sigma = \Sigma' \cup \{\mathcal{R}, \beta\}$ with $\mathcal{R}, \beta$ two new terminals of arity one.

Definition 2. For Δ a set of terminals, a Δ-term is a, not necessarily well-founded, tree labelled with elements of Δ where every node labelled with $\mathfrak{f}$ has $\sharp(\mathfrak{f})$ many children.

Example 3. Let $\Sigma' = \{\mathtt{f}, \mathtt{g}, \mathtt{a}\}$ with $\mathtt{f}$, $\mathtt{g}$ and $\mathtt{a}$ of arities 2, 1, and 0, respectively. Figure 1 shows two Σ'-terms.

Let $\mathfrak{A}$ be a fixed nondeterministic tree automaton with state set Q and transition function $\delta\colon Q \times \Sigma \to \mathfrak{P}((Q \cup \{*\})^N)$ where $N = \max\{\sharp(\mathfrak{g}) \mid \mathfrak{g} \in \Sigma\}$ is the maximal arity and $\delta(q, \mathfrak{g}) \subset Q^{\sharp(\mathfrak{g})} \times \{*\}^{N-\sharp(\mathfrak{g})}$ whenever $q \in Q$ and $\mathfrak{g} \in \Sigma$.

In other words, let $\mathfrak{A}$ be a nondeterministic tree automaton that works on Σ-terms.

Definition 4. We say that $\mathfrak{A}$ *has a run up to level* n, if it has a run that goes at least till all nodes with distance at most n from the root.

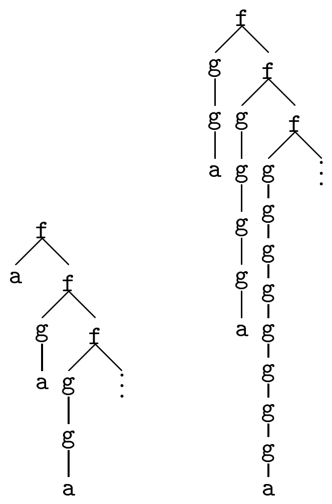


Fig. 1. Two $\{\mathtt{f}, \mathtt{g}, \mathtt{a}\}$-terms.

We write $\mathfrak{A}, q \models^n t$ to denote that $\mathfrak{A}$ has a run on t up to level n starting in state q. We write $\mathfrak{A}, q \models^\infty t$ to denote that $\mathfrak{A}$ has an *infinite* run on t starting in state q. Since there are only finitely many ways to extend a run of length n to one of length $n + 1$, by König's Lemma we obtain that $\mathfrak{A}, q \models^\infty t$ if and only if $\forall n. \mathfrak{A}, q \models^n t$.

Example 5. Continuing Example 3 consider the property

> "Every maximal chain of letters g has even length".

It can be expressed by an automaton with two states $Q = \{q_2, q_1\}$ where q_2 means that an even number of gs has been passed on the path so far, where q_1 means that the maximal chain of gs passed has odd length. Then the initial state is q_2 and the transition function is as follows.

$$\begin{array}{ll} \delta(\mathtt{f}, q_2) = \{(q_2, q_2)\} & \delta(\mathtt{f}, q_1) = \emptyset \\ \delta(\mathtt{g}, q_2) = \{(q_1, *)\} & \delta(\mathtt{g}, q_1) = \{(q_2, *)\} \\ \delta(\mathtt{a}, q_2) = \{(*, *)\} & \delta(\mathtt{a}, q_1) = \emptyset \end{array}$$

This automaton can be extended to work on Σ-trees by setting $\delta(q, \mathcal{R}) = \delta(q, \beta) = \{(q, *)\}$. Note that this automaton has an infinite run on the second tree in Figure 1, whereas it has a run only up to level 3 on the first one.

Definition 6. The *simple types*, denoted by ρ, σ, τ, are built from the base type ι by arrows $\rho \to \sigma$. The arrow associates to the right. In particular, $\overrightarrow{\rho} \to \iota$ is short for $\rho_1 \to (\rho_2 \to (\ldots(\rho_n \to \iota)\ldots))$.

Definition 7. *Infinitary simply-typed lambda-trees* over typed terminals Σ' are coinductively given by the grammar

$$r, s \ ::= \ x^\rho \mid (\lambda x^\rho t^\sigma)^{\rho \to \sigma} \mid (t^{\rho \to \sigma} s^\rho)^\sigma \mid \mathfrak{f}^{\iota \to \ldots \to \iota \to \iota}\,.$$

In other words, they are, not-necessarily well founded, trees built, in a locally type respecting way, from unary λx^ρ-nodes, binary @-nodes representing application, and leaf nodes consisting of typed variables x^ρ of type ρ and typed constants $\mathfrak{f} \in \Sigma'$ of type $\underbrace{\iota \to \ldots \to \iota}_{\sharp(\mathfrak{f})} \to \iota$.

Here λx^ρ binds free occurrences of the variable x^ρ in its body. Trees with all variables bound are called *closed*.

A lambda-tree with only finitely many non-isomorphic subtrees is called *regular*.

We omit type superscripts if they are clear from the context, or irrelevant.

We usually leave out the words “simply typed”, tacitly assuming all our lambda-trees to be simply typed and to use terminals from Σ' only. Figure 2 shows two regular lambda-trees. Arrows are used to show where the pattern repeats, or to draw isomorphic subtrees only once. Note that they denote terms (shown in Figure 1) that are not regular. Here, by “denote” we mean the *term reading* of the normal form.

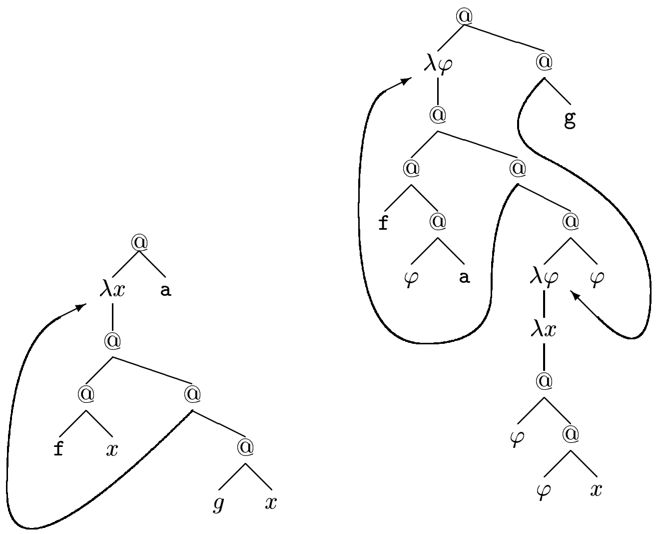
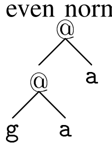


Fig. 2. Two regular lambda-trees with denotation being the $\{\mathtt{f}, \mathtt{g}, \mathtt{a}\}$-terms in Figure 1.

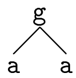
Remark 8. It should be noted that in lambda-trees, as opposed to Σ'-terms, all constants and variables, no matter what their type is, occur at leaf positions.

The reason is, that in a lambda-calculus setting the main concept is that of an application. This is different from first order terms, where the constructors are the main concept.

Note that we use lambda-trees to denote Σ'-terms. As these are different concepts, even normal lambda-trees differ from their denotation. For example the lambda-tree



denotes the Σ'-term



.

3 Recursion Schemes as Means to Define Regular Lambda Trees

The interest in infinitary lambda-trees in the verification community recently arose by the study of recursion schemes. It could be shown [4, 5] that under a certain “safety” condition the (infinite) terms generated by recursion schemes have decidable monadic second order theory. For our purpose it is enough to consider recursion schemes as a convenient means to define regular lambda-trees.

Definition 9. *Recursion schemes* are given by a set of first order terminal symbols, simply-typed non-terminal symbols and for every non-terminal F an equation

$$F\overrightarrow{x} = e$$

where e is an expression of ground type built up from terminals, non-terminals and the variables $\overrightarrow{x}$ by type-respecting application. There is a distinguished non-terminal symbol S of ground type, called the *start symbol*.

Definition 10. Each recursion scheme *denotes*, in the obvious way, a partial, in general infinite, term built from the terminals. Starting from the start symbol, recursively replace the outer-most non-terminals by their definitions with the arguments substituted in appropriately.

Definition 11. To every recursion scheme is *associated* a regular lambda-tree in the following way. First replace all equations $F\overrightarrow{x} = e$ by

$$F = \lambda\overrightarrow{x}.e$$

where the right hand side is read as a lambda term.

Then, starting from the start symbol, recursively replace all non-terminals by their definition *without performing any computations*.

Remark 12. Immediately from the definition we note that the β-normal form of the lambda-tree associated with a recursion scheme, when read a term, *is* the term denoted by that recursion scheme.

$$\begin{array}{ll} S = F\mathtt{a} \\ Fx = \mathtt{f}x(F(\mathtt{g}x)) \end{array} \qquad \begin{array}{ll} S' &= F'(W\mathtt{g}) \\ F'\varphi &= \mathtt{f}(\varphi a)(F'(W\varphi)) \\ W\varphi x &= \varphi(\varphi x) \end{array}$$

Fig. 3. Two recursion schemes.

Example 13. Figure 3 shows two recursion schemes with non-terminals $F\colon \iota \to \iota$, $F'\colon (\iota \to \iota) \to \iota$, $W\colon (\iota \to \iota) \to \iota \to \iota$, and $S, S'\colon \iota$. Their corresponding lambda-trees are the ones shown in Figure 2. The sharing of an isomorphic sub-tree arises as both are translations of the same non-terminal W. As already observed, these recursion schemes denote the terms shown in Figure 1.

Remark 14. The notion of a recursion scheme wouldn't change if we allowed λ-abstractions on the right hand side of the equations; we can always build the closure and "factor it out" as a new non-terminal. For example, the $W\varphi$ in the definition of F' in Figure 3 should be thought of as a factored-out closure of a line that originally looked

$$F'\varphi = \mathtt{f}(\varphi a)(F'(\lambda x.\varphi(\varphi x))) \ .$$

4 Using Continuous Normalisation

As mentioned in the introduction, we are interested in the question, whether $\mathfrak{A}$ has a run on the normal form of some lambda-tree t. Our plan to investigate this question is by analysing the term t.

However, there is no bound on the number of nodes of t that have to be inspected, and no bound on the number of beta-reductions to be carried out, before the first symbol of the normal form is determined — if it ever will be. In fact, it may well be that an infinite simply-typed lambda-tree leaves the normal form undefined at some point.

Whereas the first observation is merely a huge inconvenience, the second observation makes it unclear what it even is supposed to mean that "$\mathfrak{A}$ has a run on the normal form of t" — if there is no such normal form.

Fortunately, it is long known how to overcome this problem. If we don't know any definite answer yet, we just output a "don't know" constructor and carry on. This idea is known as "continuous normalisation" [7, 8] and is quite natural [1] in the realm of the lambda calculus.

Definition 15. For t, $\overrightarrow{t}$ closed infinitary simply-typed lambda-trees such that $t\overrightarrow{t}$ is of ground type we define a Σ-term $t\underline{@}\overrightarrow{t}$ coinductively as follows.

$$
\begin{aligned}
(rs)\underline{@}\overrightarrow{t} &= \mathcal{R}(r\underline{@}(s,\overrightarrow{t}))\\
(\lambda x.r)\underline{@}(s,\overrightarrow{t}) &= \beta(r[s/x]\underline{@}\overrightarrow{t})\\
\mathfrak{f}\underline{@}\overrightarrow{t} &= \mathfrak{f}(t_1^{\beta},\dots,t_n^{\beta})
\end{aligned}
$$

Here we used $r[s/x]$ to denote the substitution of s for x in r. This substitution is necessarily capture free as s is closed. By $\mathfrak{f}(T_1,\dots,T_n)$ we denote the term with label $\mathfrak{f}$ at the root and $T_1,\dots,T_n$ as its n children; this includes the case $n = 0$, where $\mathfrak{f}()$ denotes the term consisting of a single node $\mathfrak{f}$. Similar notation is used for $\mathcal{R}(T)$ and $\beta(T)$. Moreover we used r^{β} as a shorthand for $r\underline{@}()$.

Immediately from the definition we notice that, after removing the $\mathcal{R}$ and β constructors, $r\underline{@}\overrightarrow{s}$ is the term reading of the normal form of $r\overrightarrow{s}$, whenever the latter is defined.

The number of β constructors counts the number of reductions necessary to reach a particular letter of the normal form [1]. Therefore, $\mathfrak{A}$ can talk not only about properties of the normal form of t, but also about the computation that led there.

It should be noted that no price has to be paid for this extra expressivity. Given an automaton on Σ' we can extend its transition function δ by setting $\delta(q,\mathcal{R}) = \delta(q,\beta) = \{(q,*,\dots,*)\}$.

5 Finitary Semantics and Proof System

The main technical idea of this article is to use a finite semantics for the simple types, describing how $\mathfrak{A}$ "sees" an object of that type.

Definition 16. For τ a simple type we define $[\![\tau]\!]$ inductively as follows.

$$\begin{array}{ll} [\![\iota]\!] & = \mathfrak{P}(Q) \\ [\![\rho \to \sigma]\!] & = {}^{[\![\rho]\!]}[\![\sigma]\!] \end{array}$$

In other words, we start with the powerset of the state set of $\mathfrak{A}$ in the base case, and use the full set theoretic function space for arrow-types.

Remark 17. Obviously all the $[\![\tau]\!]$ are finite sets.

Example 18. Taking the automaton $\mathfrak{A}$ of Example 5, we have $[\![\iota]\!] = \{\emptyset, \{q_2\}, \{q_1\}, Q\}$ and examples of elements of $[\![\iota \to \iota]\!]$ include the identity function id, as well as the "swap function" swap defined by $\mathrm{swap}(\emptyset) = \emptyset$, $\mathrm{swap}(Q) = Q$, $\mathrm{swap}(\{q_2\}) = \{q_1\}$, and $\mathrm{swap}(\{q_1\}) = \{q_2\}$.

Definition 19. $[\![\tau]\!]$ is partially ordered as follows.

- For $R, S \in [\![\iota]\!]$ we set $R \sqsubseteq S$ iff $R \subseteq S$.
- For $f, g \in [\![\rho \to \sigma]\!]$ we set $f \sqsubseteq g$ iff $\forall a \in [\![\rho]\!].fa \sqsubseteq ga$.

Remark 20. Obviously suprema and infima with respect to $\sqsubseteq$ exist.

We often need the concept "continue with f after reading one $\mathcal{R}$ symbol". We call this $\mathcal{R}$-lifting. Similar for β.

Definition 21. For $f \in [\![\overrightarrow{\rho} \to \iota]\!]$ we define the liftings $\mathcal{R}(f), \beta(f) \in [\![\overrightarrow{\rho} \to \iota]\!]$ as follows.

$$\begin{array}{l} \mathcal{R}(f)(\overrightarrow{a}) = \{q \mid \delta(q, \mathcal{R}) \cap f\overrightarrow{a} \times \{*\} \times \ldots \times \{*\} \neq \emptyset\} \\ \beta(f)(\overrightarrow{a}) \ = \{q \mid \delta(q, \beta) \cap f\overrightarrow{a} \times \{*\} \times \ldots \times \{*\} \neq \emptyset\} \end{array}$$

Remark 22. If $\mathfrak{A}$ is obtained from an automaton working on Σ'-terms by setting $\delta(q, \mathcal{R}) = \delta(q, \beta) = \{(q, *, \ldots, *)\}$ then $\mathcal{R}(f) = \beta(f) = f$ for all f.

Using this finite semantics we can use it to annotate a lambda-tree by semantical values for its subtrees to show that the denoted term has good properties with respect to $\mathfrak{A}$. We start by an example.

Example 23. The second recursion scheme in Figure 3 denotes a term where the "side branches" contain $2, 4, 8, \ldots, 2^n, \ldots$ times the letter g. As these are all even numbers, $\mathfrak{A}$ should have a run when starting in q_2.

So we start by assigning the root $\{q_2\} \in [\![\iota]\!]$. Since the term is an application, we have to guess the semantics of the argument (of type $\iota \to \iota$). Our (correct) guess is, that it keeps the parity of gs unchanged, hence our guess is id; the function side then must be something that maps id to $\{q_2\}$. Let us denote by $\mathrm{id} \mapsto \{q_2\}$ the function in ${}^{[\![\iota \to \iota]\!]}[\![\iota]\!]$ defined by $(\mathrm{id} \mapsto \{q_2\})(\mathrm{id}) = \{q_2\}$ and $(\mathrm{id} \mapsto \{q_2\})(f) = \emptyset$ if $f \neq \mathrm{id}$.

The next node to the left is an abstraction. So we have to assign the body the value $\{q_2\}$ in a context where φ is mapped to id. Let us denote this context by Γ_φ.

In a similar way we fill out the remaining annotations. Figure 4 shows the whole proof. Here Γ'_φ is the context that maps φ to swap; moreover $\Gamma_{\varphi,x}$, $\Gamma'_{\varphi,x}$, $\Gamma_{\varphi,x'}$, and $\Gamma'_{\varphi,x'}$ are the same as Γ_φ and Γ'_φ but with x mapped to $\{q_2\}$ and $\{q_1\}$, respectively.

It should be noted that a similar attempt to assign semantical values to the other lambda-tree in Figure 2 fails at the down-most x where in the context Γ with $\Gamma(x) = \{q_2\}$ we cannot assign x the value $\{q_1\}$.

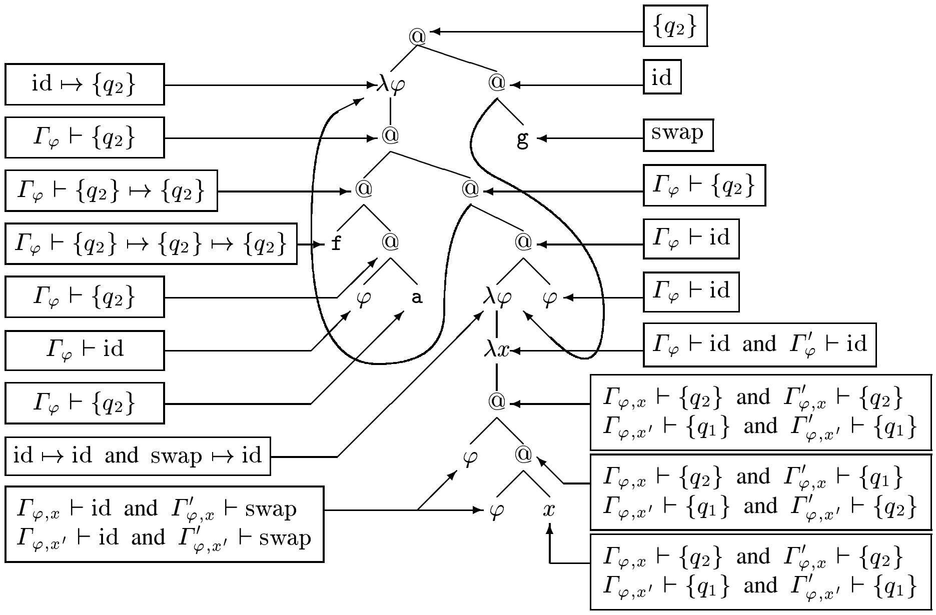


Fig. 4. A proof that $\mathfrak{A}$ has an infinite run starting in q_2 on the denoted term.

To make the intuition of the example precise, we formally define a "proof system" of possible annotations (Γ, a) for a (sub)tree. Since the $[\![\tau]\!]$ are all finite sets, there are only finitely many possible annotations.

To simplify the later argument on our proof, which otherwise would be coinductive, we add a level n to our notion of proof. This level should be interpreted as "for up to n steps we can pretend to have a proof". This reflects the fact that coinduction is nothing but induction on observations.

Definition 24. For Γ a finite mapping from variables x^σ to their corresponding semantics $[\![\sigma]\!]$, a value $a \in [\![\rho]\!]$, and t an infinitary, maybe open, lambda-tree of type ρ, with free variables among $\mathrm{dom}(\Gamma)$, we define

$$\Gamma \vdash^n_{\mathfrak{A}} a \sqsubseteq t : \rho$$

by induction on the natural number n as follows.

- $\Gamma \vdash^0_{\mathfrak{A}} a \sqsubseteq t : \rho$ always holds.
- $\Gamma \vdash^n_{\mathfrak{A}} a \sqsubseteq x_i : \rho$ holds, provided $a \sqsubseteq \Gamma(x_i)$.
- $\Gamma \vdash^{n+1}_{\mathfrak{A}} a \sqsubseteq st : \sigma$ holds, provided there exists $f \in [\![\rho \to \sigma]\!]$, $u \in [\![\rho]\!]$ such that $a \sqsubseteq \mathcal{R}(fu)$, $\Gamma \vdash^n_{\mathfrak{A}} f \sqsubseteq s : \rho \to \sigma$, and $\Gamma \vdash^n_{\mathfrak{A}} u \sqsubseteq t : \rho$.
- $\Gamma \vdash^{n+1}_{\mathfrak{A}} f \sqsubseteq \lambda x^\rho.s : \rho \to \sigma$ holds, provided for all $a \in [\![\rho]\!]$ there is a $b_a \in [\![\sigma]\!]$ such that $fa \sqsubseteq \beta(b_a)$ and $\Gamma^a_x \vdash^n_{\mathfrak{A}} b_a \sqsubseteq s : \sigma$.
- $\Gamma \vdash^n_{\mathfrak{A}} f \sqsubseteq \mathfrak{f} : \iota \to \ldots \to \iota \to \iota$ holds, provided for all $\overrightarrow{a} \in [\![\overrightarrow{\iota}]\!]$ we have $f\overrightarrow{a} \subset \{q \mid \delta(q, \mathfrak{f}) \cap a_1 \times \ldots \times a_{\sharp(\mathfrak{f})} \times \{*\} \times \ldots \times \{*\} \neq \emptyset\}$.

It should be noted that all the quantifiers in the rules range over finite sets. Hence the correctness of a rule application can be checked effectively (and even by a finite automaton).

We write $\Gamma \vdash^{\infty}_{\mathfrak{A}} a \sqsubseteq t : \rho$ to denote $\forall n. \Gamma \vdash^{n}_{\mathfrak{A}} a \sqsubseteq t : \rho$.

Remark 25. Obviously $\Gamma \vdash^{n+1}_{\mathfrak{A}} a \sqsubseteq t : \rho$ implies $\Gamma \vdash^{n}_{\mathfrak{A}} a \sqsubseteq t : \rho$. Moreover, $a' \sqsubseteq a$ and $\Gamma \vdash^{n}_{\mathfrak{A}} a \sqsubseteq t : \rho$ imply $\Gamma \vdash^{n}_{\mathfrak{A}} a' \sqsubseteq t : \rho$. Finally, $\Gamma \vdash^{n}_{\mathfrak{A}} a \sqsubseteq t : \rho$, if $\Gamma' \vdash^{n}_{\mathfrak{A}} a \sqsubseteq t : \rho$ for some Γ' which agrees with Γ on the free variables of t.

As already mentioned, for t a term with finitely many free variables, the annotations (Γ, a) come from a fixed finite set, since we can restrict Γ to the set of free variables of t. If, moreover, t has only finitely many different sub-trees, that is to say, if t is regular, then only finitely many terms t have to be considered. So we obtain

Proposition 26. *For t regular, it is decidable whether $\Gamma \vdash^{\infty}_{\mathfrak{A}} a \sqsubseteq t : \rho$.*

Before we continue and show our calculus in Definition 24 to be sound (Section 6) and complete (Section 7) let us step back and see what we will then have achieved, once our calculus is proven sound and complete.

Proposition 26 gives us decidability for terms denoted by regular lambda-trees, and hence in particular for trees obtained by recursion schemes. Moreover, since the annotations only have to fit locally, individual subtrees of the lambda-tree can be verified separately. This is of interest, as for each non-terminal a separate subtree is generated. In other words, this approach allows for modular verification; think of the different non-terminals as different subroutines. As the semantics is the set-theoretic one, the annotations are clear enough to be meaningful, if we have chosen our automaton in such a way that the individual states can be interpreted extensionally, for example as "even" versus "odd" number of gs.

It should also be noted, that the number of possible annotations only depends on the type of the subtree, and on $\mathfrak{A}$, that is, the property to be investigated. Fixing $\mathfrak{A}$ and the allowed types (which both usually tend to be quite small), the amount of work to be carried out grows only linearly with the representation of t as a regular lambda-tree. For every node we have to make a guess and we have to check whether this guess is consistent with the guesses for the (at most two) child nodes. Given that the number of nodes of the representation of t growth linearly with the size of the recursion scheme, the problem is in fixed-parameter-$\mathcal{NP}$, which doesn't seem too bad for practical applications.

6 Truth Relation and Proof of Soundness

The soundness of a calculus is usually shown by using a logical relation, that is, a relation indexed by a type that interprets the type arrow "$\rightarrow$" as logical arrow "$\Rightarrow$"; in other words, we define partial truth predicates for the individual types [11].

Since we want to do induction on the "observation depth" n of our proof $\cdot \vdash^{n}_{\mathfrak{A}} \cdot \sqsubseteq \cdot : \tau$ we have to include that depth in the definition of our truth predicates $\cdot \lll^{n}_{\mathfrak{A}} \cdot : \tau$. For technical reasons we have to build in weakening on this depth as well.

Definition 27. For $f \in [\![\overrightarrow{\rho} \to \iota]\!]$, $n \in \mathbb{N}$, t a closed infinitary lambda tree of type $\overrightarrow{\rho} \to \iota$, the relation $f \prec\!\!\!\prec^n_{\mathfrak{A}} t : \overrightarrow{\rho} \to \iota$ is defined by induction on the type as follows.

$$\begin{array}{l} f \prec\!\!\!\prec^n_{\mathfrak{A}} t : \overrightarrow{\rho} \to \iota \qquad \text{iff} \\ \forall \ell \leq n \forall \overrightarrow{a} \in [\![\overrightarrow{\rho}]\!] \forall \overrightarrow{r} : \overrightarrow{\rho} \\ \quad (\forall i.\, a_i \prec\!\!\!\prec^\ell_{\mathfrak{A}} r_i : \rho_i) \Rightarrow \forall q \in f\overrightarrow{a}.\, \mathfrak{A}, q \models^\ell t\underline{@}\overrightarrow{r} \end{array}$$

Remark 28. Immediately from the definition we get the following monotonicity property.

If $f \sqsubseteq f'$ and $f' \prec\!\!\!\prec^n_{\mathfrak{A}} t : \rho$ then $f \prec\!\!\!\prec^n_{\mathfrak{A}} t : \rho$.

Remark 29. In the special case $\overrightarrow{\rho} = \varepsilon$ we get

$$S \prec\!\!\!\prec^n_{\mathfrak{A}} t : \iota \qquad \text{iff} \qquad \forall q \in S. \mathfrak{A}, q \models^n t^\beta$$

Here we used that $\forall \ell \leq n.\, \mathfrak{A}, q \models^\ell s$ iff $\mathfrak{A}, q \models^n s$.

Immediately from the definition we obtain weakening in the level.

Proposition 30. *If $f \prec\!\!\!\prec^n_{\mathfrak{A}} t : \rho$ then $f \prec\!\!\!\prec^{n-1}_{\mathfrak{A}} t : \rho$.*

Theorem 31. *Assume $\Gamma \vdash^n_{\mathfrak{A}} a \sqsubseteq t : \rho$ for some Γ with domain $\{x_1, \ldots, x_2\}$. For all $\ell \leq n$ and all closed terms $\overrightarrow{t} : \overrightarrow{\rho}$, if $\forall i.\, \Gamma(x_i) \prec\!\!\!\prec^\ell_{\mathfrak{A}} t_i : \rho_i$ then $a \prec\!\!\!\prec^\ell_{\mathfrak{A}} t[\overrightarrow{t}/\overrightarrow{x}] : \rho$.*

Proof. Induction on n, cases according to $\Gamma \vdash^n_{\mathfrak{A}} a \sqsubseteq t : \rho$.

We just show the case of the λ-rule. The other cases are similar, and even simpler.

Case $\Gamma \vdash^{n+1}_{\mathfrak{A}} f \sqsubseteq \lambda x^\rho.s : \rho \to \sigma$ thanks to $\forall a \in [\![\rho]\!]\, \exists b_a \in [\![\sigma]\!]$ such that $fa \sqsubseteq \beta(b_a)$ and $\Gamma^a_x \vdash^n_{\mathfrak{A}} b_a \sqsubseteq s : \sigma$.

Let $\ell \leq n+1$ be given and $\overrightarrow{t} : \overrightarrow{\rho}$ with $\Gamma(x_i) \prec\!\!\!\prec^\ell_{\mathfrak{A}} t_i : \rho_i$.

We have to show $f \prec\!\!\!\prec^\ell_{\mathfrak{A}} (\lambda x^\rho s^\sigma)\eta : \rho \to \sigma$ where η is short for $[\overrightarrow{t}/\overrightarrow{x}]$.

Let σ have the form $\sigma = \overrightarrow{\sigma} \to \iota$. Let $k \leq \ell$ be given and $r : \rho$, $\overrightarrow{s} : \overrightarrow{\sigma}$, $c \in [\![\rho]\!]$, $c_i \in [\![\sigma_i]\!]$ such that $c \prec\!\!\!\prec^k_{\mathfrak{A}} r : \rho$, $c_i \prec\!\!\!\prec^k_{\mathfrak{A}} s_i : \sigma_i$. We have to show for all $q \in fc\overrightarrow{c}$ that $\mathfrak{A}, q \models^k \underbrace{(\lambda xs)\eta\underline{@}r, \overrightarrow{s}}_{\beta.s\eta^r_x\underline{@}\overrightarrow{s}}$.

Hence it suffices to show that there is a $\tilde{q} \in \delta(q, \beta)$ such that $\mathfrak{A}, \tilde{q} \models^{k-1} s\eta^r_x\underline{@}\overrightarrow{s}$.

We know $c \prec\!\!\!\prec^k_{\mathfrak{A}} r : \rho$; using Proposition 30 we get $c \prec\!\!\!\prec^{k-1}_{\mathfrak{A}} r : \rho$ and $\forall i.\, \Gamma(x_i) \prec\!\!\!\prec^{k-1}_{\mathfrak{A}} t_i : \rho_i$. Since $k \leq \ell \leq n+1$ we get $k-1 \leq n$, hence we may apply the induction hypothesis to $\Gamma^a_x \vdash^n_{\mathfrak{A}} b_a \sqsubseteq s : \sigma$ and obtain $b_a \prec\!\!\!\prec^{k-1}_{\mathfrak{A}} s\eta^r_x : \sigma$.

Since again by Proposition 30 we also know $c_i \prec\!\!\!\prec^{k-1}_{\mathfrak{A}} s_i : \sigma_i$, we obtain for all $\hat{q} \in b_a\overrightarrow{c}$ that $\mathfrak{A}, \hat{q} \models^{k-1} s\eta^r_x\underline{@}\overrightarrow{s}$.

Since $fc \sqsubseteq \beta(b_c)$ we get that $\forall q \in fc\overrightarrow{c}\, \exists \tilde{q} \in \delta(q, \beta).\, \tilde{q} \in b_c\overrightarrow{c}$. This, together with the last statement yields the claim.

It should be noted that in the proof of Theorem 31 in the cases of the λ-rule and the application-rule it was possible to use the induction hypothesis due to the fact that we used *continuous* normalisation, as opposed to standard normalisation.

Corollary 32. *For t a closed infinitary lambda term we get immediately from Theorem 31*

$$\emptyset \vdash^n_{\mathfrak{A}} S \sqsubseteq t : \iota \quad \Longrightarrow \quad \forall q \in S.\, \mathfrak{A}, q \models^n t^\beta$$

In particular, if $\emptyset \vdash^\infty_{\mathfrak{A}} S \sqsubseteq t : \iota$ then $\forall q \in S.\, \mathfrak{A}, q \models^\infty t^\beta$.

7 The Canonical Semantics and the Proof of Completeness

If we want to prove that there is an infinite run, then, in the case of an application st, we have to guess a value for the term t "cut out".

We could assume an actual run be given and analyse the "communication", in the sense of game semantics [3], between the function s and its argument t. However, it is simpler to assign each term a "canonical semantics" $\langle\!\langle t\rangle\!\rangle_{\mathfrak{A}\infty}$, roughly the supremum of all values we have canonical proofs for.

The subscript ∞ signifies that we only consider infinite runs. The reason is that the level n in our proofs $\Gamma \vdash^n_{\mathfrak{A}} a \sqsubseteq t : \rho$ is not a tight bound; whenever we have a proofs of level n, then there are runs for at least n steps, but on the other hand, runs might be longer than the maximal level of a proof. This is due to the fact that β-reduction moves subterms "downwards", that is, further away from the root, and in that way may construct longer runs. The estimates in our proof calculus, however, have to consider (in order to be sound) the worst case, that is, that an argument is used immediately.

Since, in general, the term t may also have free variables, we have to consider a canonical semantics $\langle\!\langle t\rangle\!\rangle^{\Gamma}_{\mathfrak{A}\infty}$ with respect to an environment Γ.

Definition 33. By induction on the type we define for t a closed infinite lambda-tree of type $\rho = \overrightarrow{\rho} \to \iota$ its canonical semantics $\langle\!\langle t\rangle\!\rangle_{\mathfrak{A}\infty} \in [\![\rho]\!]$ as follows.

$$\langle\!\langle t\rangle\!\rangle_{\mathfrak{A}\infty}(\overrightarrow{a}) = \{q \mid \exists \overrightarrow{s} : \overrightarrow{\rho}\ .\ \langle\!\langle \overrightarrow{s}\rangle\!\rangle_{\mathfrak{A}\infty} \sqsubseteq \overrightarrow{a}\ \wedge \mathfrak{A}, q \models^{\infty} t\underline{@}\overrightarrow{s}\}$$

Remark 34. For t a closed term of base type we have $\langle\!\langle t\rangle\!\rangle_{\mathfrak{A}\infty} = \{q \mid \mathfrak{A}, q \models^{\infty} t^{\beta}\}$.

Definition 35. For Γ a context, $t\colon \rho$ typed in context Γ of type $\rho = \overrightarrow{\rho} \to \iota$ we define $\langle\!\langle t\rangle\!\rangle^{\Gamma}_{\mathfrak{A}\infty} \in [\![\rho]\!]$ by the following explicit definition.

$$\begin{aligned}\langle\!\langle t\rangle\!\rangle^{\Gamma}_{\mathfrak{A}\infty}(\overrightarrow{a}) = \{q \mid\ & \exists \eta.\ \mathrm{dom}(\eta) = \mathrm{dom}(\Gamma) \wedge \\ & (\forall x \in \mathrm{dom}(\Gamma).\eta(x) \text{ closed} \wedge \langle\!\langle \eta(x)\rangle\!\rangle_{\mathfrak{A}\infty} \sqsubseteq \Gamma(x)) \wedge \\ & \exists \overrightarrow{s} : \overrightarrow{\rho}.\langle\!\langle \overrightarrow{s}\rangle\!\rangle_{\mathfrak{A}\infty} \sqsubseteq \overrightarrow{a}\ \wedge\ \mathfrak{A}, q \models^{\infty} t\eta\underline{@}\overrightarrow{s}\}\end{aligned}$$

Remark 36. For t a closed term and $\Gamma = \emptyset$ we have $\langle\!\langle t\rangle\!\rangle^{\Gamma}_{\mathfrak{A}\infty} = \langle\!\langle t\rangle\!\rangle_{\mathfrak{A}\infty}$.

Proposition 37. *If s has type $\overrightarrow{\sigma} \to \iota$ in some context compatible with Γ, and η is some substitution with $\mathrm{dom}(\eta) = \mathrm{dom}(\Gamma)$ such that for all $x \in \mathrm{dom}(\Gamma)$ we have $\eta(x)$ closed and $\langle\!\langle \eta(x)\rangle\!\rangle_{\mathfrak{A}\infty} \sqsubseteq \Gamma(x)$, then*

$$\langle\!\langle s\eta\rangle\!\rangle_{\mathfrak{A}\infty} \sqsubseteq \langle\!\langle s\rangle\!\rangle^{\Gamma}_{\mathfrak{A}\infty}$$

Proof. Let $\overrightarrow{a} \in [\![\overrightarrow{\sigma}]\!]$ and $q \in \langle\!\langle s\eta\rangle\!\rangle_{\mathfrak{A}\infty}(\overrightarrow{a})$ be given. Then there are $\overrightarrow{s} : \overrightarrow{\sigma}$ with $\langle\!\langle \overrightarrow{s}\rangle\!\rangle_{\mathfrak{A}\infty} \sqsubseteq \overrightarrow{a}$ such that $\mathfrak{A}, q \models^{\infty} s\eta\underline{@}\overrightarrow{s}$. Together with the assumed properties of η this witnesses $q \in \langle\!\langle s\rangle\!\rangle^{\Gamma}_{\mathfrak{A}\infty}(\overrightarrow{a})$.

Lemma 38. *If r and s are terms of type $\sigma \to \overrightarrow{\rho} \to \iota$ and σ, respectively, in some context compatible with Γ, then we have*

$$\langle\!\langle rs\rangle\!\rangle^{\Gamma}_{\mathfrak{A}\infty} \sqsubseteq \mathcal{R}(\langle\!\langle r\rangle\!\rangle^{\Gamma}_{\mathfrak{A}\infty}\langle\!\langle s\rangle\!\rangle^{\Gamma}_{\mathfrak{A}\infty})$$

Proof. Let $\overrightarrow{a} \in [\![\overrightarrow{\rho}]\!]$ and $q \in \langle\!\langle rs \rangle\!\rangle^{\Gamma}_{\mathfrak{A}\infty}(\overrightarrow{a})$ be given. Then there is η with $\forall x \in \mathrm{dom}(\Gamma)$. $\langle\!\langle \eta(x) \rangle\!\rangle_{\mathfrak{A}\infty} \sqsubseteq \Gamma(x)$ and there are $\overrightarrow{s} : \overrightarrow{\rho}$ with $\langle\!\langle \overrightarrow{s} \rangle\!\rangle_{\mathfrak{A}\infty} \sqsubseteq \overrightarrow{a}$ and

$$\mathfrak{A}, q \models^{\infty} \underbrace{(rs)\eta \underline{@} \overrightarrow{s}}_{\mathcal{R}.r\eta\underline{@}s\eta,\overrightarrow{s}}$$

Hence there is a $q' \in \delta(q, \mathcal{R})$ with $\mathfrak{A}, q' \models^{\infty} r\eta\underline{@}s\eta, \overrightarrow{s}$. It suffices to show that for this q' we have $q' \in \langle\!\langle r \rangle\!\rangle^{\Gamma}_{\mathfrak{A}\infty} \langle\!\langle s \rangle\!\rangle^{\Gamma}_{\mathfrak{A}\infty} \overrightarrow{a}$.

By Proposition 37 we have $\langle\!\langle s\eta \rangle\!\rangle_{\mathfrak{A}\infty} \sqsubseteq \langle\!\langle s \rangle\!\rangle^{\Gamma}_{\mathfrak{A}\infty}$ and we already have $\langle\!\langle \overrightarrow{s} \rangle\!\rangle_{\mathfrak{A}\infty} \sqsubseteq \overrightarrow{a}$. So the given η together with $s\eta$ and $\overrightarrow{s}$ witnesses $q' \in \langle\!\langle r \rangle\!\rangle^{\Gamma}_{\mathfrak{A}\infty} \langle\!\langle s \rangle\!\rangle^{\Gamma}_{\mathfrak{A}\infty} \overrightarrow{a}$.

Lemma 39. *Assume that $\lambda x.r$ has type $\sigma \to \overrightarrow{\rho} \to \iota$ in some context compatible with Γ. Then*

$$\langle\!\langle \lambda xr \rangle\!\rangle^{\Gamma}_{\mathfrak{A}\infty}(a) \sqsubseteq \beta(\langle\!\langle r \rangle\!\rangle^{\Gamma^a_x}_{\mathfrak{A}\infty})$$

Proof. Let $\overrightarrow{a} \in [\![\overrightarrow{\rho}]\!]$ and $q \in \langle\!\langle \lambda xr \rangle\!\rangle^{\Gamma}_{\mathfrak{A}\infty}(a, \overrightarrow{a})$ be given. Then there is an η with $\forall x \in \mathrm{dom}(\Gamma)$ we have $\eta(x)$ closed and $\langle\!\langle \eta(x) \rangle\!\rangle_{\mathfrak{A}\infty} \sqsubseteq \Gamma(x)$ and there are $s, \overrightarrow{s}$ with $\langle\!\langle s \rangle\!\rangle_{\mathfrak{A}\infty} \sqsubseteq a$ and $\langle\!\langle \overrightarrow{s} \rangle\!\rangle_{\mathfrak{A}\infty} \sqsubseteq \overrightarrow{a}$ such that

$$\mathfrak{A}, q \models^{\infty} \underbrace{(\lambda xr)\eta \underline{@} s, \overrightarrow{s}}_{\beta.r_x[s]\eta\underline{@}\overrightarrow{s}}$$

So there is a $\tilde{q} \in \delta(q, \beta)$ with $\mathfrak{A}, \tilde{q} \models^{\infty} r_x[s]\eta\underline{@}\overrightarrow{s}$. It suffices to show that $\tilde{q} \in \langle\!\langle r \rangle\!\rangle^{\Gamma^a_x}_{\mathfrak{A}\infty}(\overrightarrow{a})$.

By the properties of η and since $\langle\!\langle s \rangle\!\rangle_{\mathfrak{A}\infty} \sqsubseteq a$ we know that for all $y \in \mathrm{dom}(\Gamma^a_x)$ we have $\langle\!\langle \eta(y) \rangle\!\rangle_{\mathfrak{A}\infty} \sqsubseteq \Gamma^a_x(y)$. This witnesses $\tilde{q} \in \langle\!\langle r \rangle\!\rangle^{\Gamma^a_x}_{\mathfrak{A}\infty}(\overrightarrow{a})$.

Lemma 40. $\langle\!\langle x \rangle\!\rangle^{\Gamma}_{\mathfrak{A}\infty} \sqsubseteq \Gamma(x)$

Theorem 41. $\Gamma \vdash^n_{\mathfrak{A}} \langle\!\langle t \rangle\!\rangle^{\Gamma}_{\mathfrak{A}\infty} \sqsubseteq t : \rho$

Proof. Induction on n, cases on t. Trivial for $n = 0$. So let $n > 0$. We distinguish cases according to t. The cases rs, $\lambda x.r$ and x are immediately from the induction hypotheses and Lemmata 38, 39, and 40, respectively.

So, let $t = \mathfrak{f}$ be a terminal symbol. We have to show $\Gamma \vdash^n_{\mathfrak{A}} \langle\!\langle \mathfrak{f} \rangle\!\rangle^{\Gamma}_{\mathfrak{A}\infty} \sqsubseteq \mathfrak{f} : \iota \to \iota$.

So, let $\overrightarrow{S} \in [\![\overrightarrow{\iota}]\!]$ and $q \in \langle\!\langle \mathfrak{f} \rangle\!\rangle^{\Gamma}_{\mathfrak{A}\infty}(S)$. Hence there is are $\overrightarrow{s}$ of type ι with $\langle\!\langle s_i \rangle\!\rangle_{\mathfrak{A}\infty} \sqsubseteq S_i$ and $\mathfrak{A}, q \models^{\infty} \underbrace{\mathfrak{f}\underline{@}\overrightarrow{s}}_{\mathfrak{f}(\overrightarrow{s^{\beta}})}$.

So there is $(\tilde{q}_1, \ldots, \tilde{q}_{\sharp(\mathfrak{f})}, *, \ldots, *) \in \delta(q, \mathfrak{f})$ with $\mathfrak{A}, \tilde{q}_i \models^{\infty} s_i^{\beta}$. But then $\tilde{q}_i \in \langle\!\langle s_i \rangle\!\rangle_{\mathfrak{A}\infty} \subset S_i$.

Corollary 42. *If $t \colon \iota$ is closed and of ground type then $\emptyset \vdash^n_{\mathfrak{A}} \{q \mid \mathfrak{A}, q \models^{\infty} t^{\beta}\} \sqsubseteq t : \iota$.*

Finally, let us sum up what we have achieved.

Corollary 43. *For t a closed regular lambda term, and $q_0 \in Q$ it is decidable whether $\mathfrak{A}, q_0 \models^{\infty} t^{\beta}$.*

Proof. By Proposition 26 it suffices to show that $\emptyset \vdash^{\infty}_{\mathfrak{A}} \{q_0\} \sqsubseteq t : \iota$ holds, if and only if $\mathfrak{A}, q_0 \models^{\infty} t^{\beta}$.

The "if"-direction follows from Corollary 42 and the weakening provided by Remark 25. The "only if"-direction is provided by Corollary 32.

8 Model Checking

Formulae of Monadic Second Order Logic can be presented [10] by appropriate tree automata. As mentioned, we consider here only a special case. More precisely, let φ be a property that can be recognised by a non-deterministic tree automaton with trivial acceptance condition, that is, an automaton accepting by having an infinite run. In other words, let φ be such that there is an automaton $\mathfrak{A}_\varphi$ such that $\mathcal{T} \models \varphi \;\Leftrightarrow\; \mathfrak{A}_\varphi, q_0 \models^\infty \mathcal{T}$ holds for every Σ'-tree $\mathcal{T}$.

Applying the theory developed above to this setting we obtain the following.

Theorem 44. *Given a tree $\mathcal{T}$ defined by an arbitrary recursion scheme (of arbitrary level) and a property φ that can be recognised by an automaton with trivial acceptance condition it is decidable whether $\mathcal{T} \models \varphi$.*

Proof. Let t be the infinite lambda-tree associated with the recursion scheme. Then t is effectively given as a regular closed lambda term of ground type and $\mathcal{T}$ is the normal form of t.

Let $\mathfrak{A}_\varphi$ be the automaton (with initial state q_0) describing φ. By keeping the state when reading a $\mathcal{R}$ or β it can be effectively extended to an automaton $\mathfrak{A}$ that works on the continuous normal form, rather than on the usual one. So $\mathcal{T} \models \varphi \Leftrightarrow \mathfrak{A}, q_0 \models^\infty t^\beta$. The latter, however, is decidable by Corollary 43.

Remark 45. As discussed after Proposition 26 the complexity is fixed-parameter non-deterministic linear time in the size of the recursion scheme, if we consider φ and the allowed types as a parameter.

References

1. K. Aehlig and F. Joachimski. On continuous normalization. In *Proceedings of the Annual Conference of the European Association for Computer Science Logic (CSL '02)*, volume 2471 of *Lecture Notes in Computer Science*, pages 59–73. Springer Verlag, 2002.
2. K. Aehlig, J. G. d. Miranda, and C. H. L. Ong. The monadic second order theory of trees given by arbitrary level-two recursion schemes is decidable. In P. Urzyczyn, editor, *Proceedings of the 7th International Conference on Typed Lambda Calculi and Applications (TLCA '05)*, volume 3461 of *Lecture Notes in Computer Science*, pages 39–54. Springer-Verlag, Apr. 2005.
3. J. M. E. Hyland and C.-H. L. Ong. On full abstraction for PCF. *Information and Computation*, 163(2):285–408, Dec. 2000.
4. T. Knapik, D. Niwiński, and P. Urzyczyn. Deciding monadic theories of hyperalgebraic trees. In S. Abramsky, editor, *Proceedings of the 5th International Conference on Typed Lambda Caculi and Applications (TLCA '01)*, volume 2044 of *Lecture Notes in Computer Science*, pages 253–267. Springer Verlag, 2001.
5. T. Knapik, D. Niwiński, and P. Urzyczyn. Higher-order pushdown trees are easy. In M. Nielson, editor, *Proceedings of the 5th International Conference Foundations of Software Science and Computation Structures (FOSSACS '02)*, volume 2303 of *Lecture Notes in Computer Science*, pages 205–222, Apr. 2002.

6. T. Knapik, D. Niwiński, P. Urzyczyn, and I. Walukiewicz. Unsafe grammars, panic automata, and decidability. In L. Caires, G. F. Italiano, L. Monteiro, C. Palamidessi, and M. Yung, editors, *32nd International Colloquium on Automata, Languages and Programming (ICALP '05)*, volume 3580 of *Lecture Notes in Computer Science*, pages 1450–1461. Springer Verlag, 2005.
7. G. Kreisel, G. E. Mints, and S. G. Simpson. The use of abstract language in elementary metamathematics: Some pedagogic examples. In R. Parikh, editor, *Logic Colloquium*, volume 453 of *Lecture Notes in Mathematics*, pages 38–131. Springer Verlag, 1975.
8. G. E. Mints. Finite investigations of transfinite derivations. *Journal of Soviet Mathematics*, 10:548–596, 1978. Translated from: Zap. Nauchn. Semin. LOMI 49 (1975). Cited after Grigori Mints. *Selected papers in Proof Theory*. Studies in Proof Theory. Bibliopolis, 1992.
9. C.-H. L. Ong. On model-checking trees generated by higher-order recursion schemes. In *Proceedings of the Twentyfrist Annual IEEE Symposium on Logic in Computer Science (LICS '06)*, 2006. to appear.
10. M. O. Rabin. Decidability of second-order theories and automata on infinite trees. *Transactions of the American Mathematical Society*, 141:1–35, July 1969.
11. W. W. Tait. Intensional interpretations of functionals of finite type. *The Journal of Symbolic Logic*, 32(2):198–212, 1967.