Interaction Equivalence

BENIAMINO ACCATTOLI, Inria - École Polytechnique, France
ADRIENNE LANCELOT, Inria - École Polytechnique - Université Paris Cité, France
GIULIO MANZONETTO, Université Paris Cité, France
GABRIELE VANONI, Université Paris Cité, France

Contextual equivalence is the *de facto* standard notion of program equivalence. A key theorem is that contextual equivalence is an *equational theory*. Making contextual equivalence more intensional, for example taking into account the time cost of the computation, seems a natural refinement. Such a change, however, does *not* induce an equational theory, for an apparently essential reason: cost is not invariant under reduction.

In the paradigmatic case of the untyped λ-calculus, we introduce *interaction equivalence*. Inspired by game semantics, we observe the number of interaction steps between terms and contexts but—crucially—ignore their internal steps. We prove that interaction equivalence is an equational theory and characterize it as $\mathcal{B}$, the well-known theory induced by Böhm tree equality. It is the first observational characterization of $\mathcal{B}$ obtained *without* enriching the discriminating power of contexts with extra features such as non-determinism. To prove our results, we develop interaction-based refinements of the Böhm-out technique and of intersection types.

CCS Concepts: • **Theory of computation** → **Lambda calculus**; **Denotational semantics**.

Additional Key Words and Phrases: Lambda calculus, program equivalences, Böhm trees.

ACM Reference Format:
Beniamino Accattoli, Adrienne Lancelot, Giulio Manzonetto, and Gabriele Vanoni. 2025. Interaction Equivalence. *Proc. ACM Program. Lang.* 9, POPL, Article 55 (January 2025), 30 pages. https://doi.org/10.1145/3704891

1 Introduction

A cornerstone of the theory of programming languages is the acceptance of contextual equivalence as a meaningful notion—if not as *the* notion—of program equivalence. Introduced by Morris [1968] to study the untyped λ-calculus, contextual equivalence has the advantage that its definition is almost language-agnostic. Given a language $\mathcal{L}$, one simply needs the notion of contexts C of $\mathcal{L}$, usually defined as terms with a single occurrence of a special additional constant $\langle\cdot\rangle$ (the *hole* of the context), and a chosen predicate $\Downarrow$ of *observation* for $\mathcal{L}$, typically some notion of termination, which is the only language-specific ingredient. Then, contextual equivalence is defined as:

$$t \equiv^{\mathrm{ctx}} u \quad \text{if for all contexts } C\,. \quad [\, C\langle t\rangle \Downarrow \quad \Leftrightarrow \quad C\langle u\rangle \Downarrow\,]$$

Contextual equivalence embodies a *black box* behavioral principle with respect to termination. Nothing is said about the structure of t and u, it is only prescribed that any use of t in a larger program can be replaced by u (and vice-versa) without affecting the observables. One of the most studied contextual equivalences is *head contextual equivalence* $\equiv_{\mathsf{h}}^{\mathrm{ctx}}$ for the untyped λ-calculus [Barendregt 1984], where one observes termination of head reduction $\to_{\mathsf{h}}$. Another contextual equivalence

Authors' Contact Information: Beniamino Accattoli, Inria - École Polytechnique, Palaiseau, France, beniamino.accattoli@inria.fr; Adrienne Lancelot, Inria - École Polytechnique - Université Paris Cité, Palaiseau, France, lancelot@irif.fr; Giulio Manzonetto, Université Paris Cité, Paris, France, gmanzone@irif.fr; Gabriele Vanoni, Université Paris Cité, Paris, France, gabriele.vanoni@irif.fr.

ACM 2475-1421/2025/1-ART55
https://doi.org/10.1145/3704891

that was studied at length is the one of Plotkin's PCF [1977], where the observation is termination on the same value of ground type, typically a natural number. Both played crucial roles in the historical development of denotational semantics of programming languages.

Contextual equivalence is also relevant in more applied settings. The typical example being program transformations at work in compilers, for which contextual equivalence guarantees a strong form of soundness, see, for instance, the survey by Patrignani et al. [2019].

Equational Contextual Equivalence. In order for contextual equivalence $\equiv^{\mathrm{ctx}}$ to be a proper semantical notion, one actually has to show that it is an *equational theory* for $\mathcal{L}$. This means that the following two properties must be satisfied:

(1) *Compatibility*: $\equiv^{\mathrm{ctx}}$ is stable under context closure, that is, $t \equiv^{\mathrm{ctx}} u$ implies $C\langle t\rangle \equiv^{\mathrm{ctx}} C\langle u\rangle$ for all contexts C, and;
(2) *Invariance*: $\equiv^{\mathrm{ctx}}$ includes β-conversion $=_\beta$, that is, if $t =_\beta u$ then $t \equiv^{\mathrm{ctx}} u$.

While compatibility holds by definition, invariance is a non-trivial theorem, because of the universal quantification on contexts in $\equiv^{\mathrm{ctx}}$, which is notoriously hard to manage. For head contextual equivalence $\equiv^{\mathrm{ctx}}_{\mathsf{h}}$, for instance, a rewriting-based proof of the invariance property requires to prove the confluence of β-reduction and various non-trivial properties of head reduction[1].

An equational theory for $\mathcal{L}$ can be seen as a *semantics* for $\mathcal{L}$. The abstract requirement defining denotational models is that their induced equivalence[2] is an equational theory. Notably, head contextual equivalence $\equiv^{\mathrm{ctx}}_{\mathsf{h}}$ is exactly the equational theory of Scott's $\mathcal{D}_\infty$ model [1972], the first model of the untyped λ-calculus.

Termination vs Time Cost, Equationally. It is natural to wonder whether contextual equivalence can be refined by replacing the termination predicate $\Downarrow$ with a finer one $\Downarrow^k$ indicating termination in k evaluation steps. Intuitively, the number of evaluation steps is taken as a time cost model. With the compiler analogy in mind, it is indeed natural to ask that a program transformation preserves the time behavior. This refinement is exactly Sands' notion of *cost equivalence*, the symmetric form of his more famous *improvement preorder* [1996a; 1996b; 1999]:

$$t \equiv^{\mathrm{cost}} u \quad \text{if for all contexts } C, \exists k \geq 0\,. \quad [\, C\langle t\rangle \Downarrow^k \quad \Leftrightarrow \quad C\langle u\rangle \Downarrow^k \,]$$

Cost equivalence falls short, however, of being a refinement of contextual equivalence. The price to pay for the added quantitative information is that it can no longer be seen as a semantics, that is, it is *not* an equational theory. The reason is both simple and deep. It is simple because (in, say, the λ-calculus) if $t \to_\beta u$, then $C\langle u\rangle$ might take one step less to terminate than $C\langle t\rangle$, so β-conversion $=_\beta$ cannot be included in cost equivalence $\equiv^{\mathrm{cost}}$, breaking the invariance of equational theories.

Internal and External. The issue is deep as it reflects a more general tension between quantitative intensional properties such as time cost, which by their nature cannot be invariant under evaluation, and semantical notions such as equational theories, that are expected to hide the computational process. More generally, one might identify two perspectives on programs. The *internal view* studies programs *in isolation*, and it is concerned with qualitative properties such as, *e.g.*, termination, confluence, productivity, or quantitative properties such as time or space cost. The *external view*, instead, studies how programs *interact* with other programs, as it is the case for contextual equivalences, labelled transition systems (LTSs), or process calculi. Usually, the external view hides the internal dynamics of programs, for instance via the invariance requirement for contextual equivalence, or via the silent τ transitions of LTSs.

[1] For an overview of such a proof, see Appendix A of the longer version on arXiv [Accattoli et al. 2024].
[2] The equivalence induced by a model $\mathcal{M}$ is defined as $t =_{\mathcal{M}} u$ if $[\![t]\!] = [\![u]\!]$, where $[\![\cdot]\!]$ is the interpretation in the model.

The mentioned issue with cost equivalence is then an instance of a more general question: is there a way of analyzing quantitative properties externally, without interference from the hidden internal dynamics? Concretely, is there a way of measuring the amount of *interactions* between a term and its context *modulo* the internal dynamics of the term and of the context?

Interaction Equivalence and the Checkers Calculus. In this paper, we provide a positive answer. The literature on improvements has focused on application-oriented λ-calculi based on call-by-need evaluation. Our focus is different, more theoretical, we are rather interested in the semantical aspect and the understanding of the internal/external dilemma. Therefore, we place ourselves in a minimalistic setting, namely the ordinary (call-by-name) untyped λ-calculus, the denotational semantics of which has been studied in depth—see the classic book [Barendregt 1984] and its recent extension [Barendregt and Manzonetto 2022]—and aim at refining head contextual equivalence.

We introduce a new notion of cost equivalence, called *(head) interaction equivalence*, whose key property is being—as we prove—an equational theory, in contrast to what happens for cost equivalence. This is obtained via a reconciliation of the internal and the external perspectives, loosely inspired by *game semantics* [Abramsky et al. 2000; Hyland and Ong 2000].

Interaction equivalence is based on the new *checkers calculus* $\Lambda_{\circ\bullet}$, a λ-calculus enriched with two players, black $\bullet$ and white $\circ$. The idea is to duplicate the abstraction and application constructors of the λ-calculus as to have white ($\lambda_\circ x.t$ and $t \circ u$) and black variants ($\lambda_\bullet x.t$ and $t \bullet u$) of each. Next, borrowing from LTSs, one defines two variants of β-reduction, the silent one (internal to a player) and the interaction (external) one, depending on whether the constructors involved in the redex belong to the same player or not:

Silent β			Interaction β		
$(\lambda_\bullet x.t) \bullet u$	$\to_{\beta_\tau}$	$t\{x := u\}$	$(\lambda_\circ x.t) \bullet u$	$\to_{\beta_{\textcircled{\bullet}}}$	$t\{x := u\}$
$(\lambda_\circ x.t) \circ u$	$\to_{\beta_\tau}$	$t\{x := u\}$	$(\lambda_\bullet x.t) \circ u$	$\to_{\beta_{\textcircled{\bullet}}}$	$t\{x := u\}$

Interaction equivalence of t and u is then defined in the checkers calculus exactly as cost equivalence *except* that one uses the predicate $\Downarrow^{\textcircled{\bullet} k}_{\mathsf{h}_{\circ\bullet}}$ holding when checkers head reduction terminates using k interaction steps $\to_{\beta_{\textcircled{\bullet}}}$ and—crucially—*arbitrarily many* (possibly zero) silent steps $\to_{\beta_\tau}$.

Checkers interaction equivalence is then transferred to the ordinary λ-calculus via a *paint-and-wash* construction: two ordinary λ-terms t and u are interaction equivalent when their uniformly, say, black-painting $\overline{t}^{\bullet}$ and $\overline{u}^{\bullet}$ are interaction equivalent in the checkers calculus, *i.e.*:

$$t \equiv^{\mathrm{int}} u \quad \text{if for all checkers contexts } C, \exists k \geq 0\,. \quad [\, C\langle \overline{t}^{\bullet} \rangle \Downarrow^{\textcircled{\bullet} k}_{\mathsf{h}_{\circ\bullet}} \quad \Leftrightarrow \quad C\langle \overline{u}^{\bullet} \rangle \Downarrow^{\textcircled{\bullet} k}_{\mathsf{h}_{\circ\bullet}} \,]$$

Our first main result is that interaction equivalence is an equational theory of the ordinary untyped λ-calculus. As for contextual equivalence, proving that $\equiv^{\mathrm{int}}$ is an equational theory is non-trivial, and the counting of interaction steps adds a further difficulty. We do it via a *multi type system*, as explained after our second contribution. We also prove that two preorder variants of interaction equivalence, one of which is the interaction reformulation of Sands's improvements, are *in*equational theories, that is, preorders verifying the compatibility and invariance of equational theories.

Inspecting Black Boxes. Contextual equivalences, as already mentioned, are black-box principles. While the concept is natural, it is hard to establish the contextual equivalence of two given terms, because of the universal quantification over all the contexts. It is then common to look for alternative, more explicit reformulations that relate the inner structure of equivalent terms. Historically, these reformulations were presented via semantic trees such as *Böhm trees*, introduced by Barendregt [1977]. Head contextual equivalence $\equiv^{\mathrm{ctx}}_{\mathsf{h}}$ for the untyped λ-calculus is one of the few cases for which an explicit description is available, as they are not easy to obtain. Namely, $\equiv^{\mathrm{ctx}}_{\mathsf{h}}$ was characterized by Hyland [1976] and Wadsworth [1976] as the equational theory $\mathcal{B}\eta^{\infty}$ induced by the equality of

Böhm trees up to possibly infinite η-equivalence. A natural question then arises: is there an explicit description of interaction equivalence? Does it correspond to anything already known?

Interaction Equivalence and Böhm Trees. The second contribution of our work is the answer to these two questions. We prove that interaction equivalence is exactly the well-known equational theory $\mathcal{B}$ induced by Böhm tree equality. In particular, interaction equivalence is *not* extensional, that is, it does not validate any form of η-equivalence, given that x and $\lambda y.xy$ clearly have a different number of interactions with any context providing an argument. Therefore, interaction equivalence $\equiv^{\text{int}}$ has a simpler explicit description than head contextual equivalence $\equiv^{\text{ctx}}_{\text{h}}$, and provides an operational insight about η-equivalence. The failure of η-equivalence also reveals that, despite the analogy, our framework is not that close to game semantics, where η is naturally validated unless additional constraints are imposed to avoid it, as in [Ker et al. 2003; Ong and Di Gianantonio 2002].

Our results actually turn out to solve an open problem in the semantical theory of the untyped λ-calculus, namely the problem of finding a satisfying description of (non-extensional) Böhm tree equality $\mathcal{B}$ as an observational equivalence. Partial solutions have been presented in the literature and are discussed among related works, in Section 9. Ours is the first solution that does not require adding any extra features such as concurrency or non-determinism, but only refining the analysis.

Normal Form Bisimilarities. Some readers might prefer a coinductive formulation of the problem. The equational theory $\mathcal{B}$, indeed, corresponds to *head normal form bisimilarity*, as shown by Lassen [1999], building over Sangiorgi's work [1994]. Normal form bisimilarities are known to be sound but—in general—*not* complete for contextual equivalences. The open problem from the literature then might be recast as follows: is there an observational equivalence for which normal form bisimilarities are complete? We show that, for the head case, interaction equivalence is the answer.

Technical Development. Our results are proved via new interaction-based refinements of standard proof methods in the literature, namely the Böhm-out technique and multi types, overviewed in the next subsection. While the proofs are non-trivial, we believe that they are compact and neat, hopefully reassuring that the introduced framework is not *ad-hoc*.

Proofs. Some proofs are omitted; they are in the proof appendix on arXiv [Accattoli et al. 2024].

Overview of the Proof Techniques

In the study of program equivalences, proving the inclusion of an equivalence into another one is always challenging. As it is customary, we study *preorders* rather than equivalences, and prove the equality of the Böhm preorder $\sqsubseteq_{\mathcal{B}}$ and the interaction preorder $\sqsubseteq^{\text{int}}$ by showing the two inclusions.

Proof Technique 1: Böhm-Out. For $\sqsubseteq^{\text{int}} \subseteq \sqsubseteq_{\mathcal{B}}$, we prove the contrapositive: if t and u have different Böhm trees then they are not interaction equivalent. We adapt the *Böhm-out technique* at work in Böhm's separation theorem [1968]—a classic result of the untyped λ-calculus—thus building a context that separates t and u. The original technique, used also more recently in [Barendregt and Manzonetto 2022; Boudol and Laneve 1996; Dezani-Ciancaglini et al. 1998; Intrigila et al. 2019], cannot distinguish η-equivalent terms. We refine it by counting interaction steps: when interacting with a context, η-equivalent terms give rise to different amounts of interaction.

Proof Technique 2: Multi Types. For the inclusion $\sqsubseteq_{\mathcal{B}} \subseteq \sqsubseteq^{\text{int}}$, we use a different technique. The main tool is a new *multi type* system (also known as non-idempotent intersection types) for the checkers calculus. Multi types are an established tool for the study of untyped λ-calculi, mediating between operational and denotational studies; see [Bucciarelli et al. 2017] for an introduction.

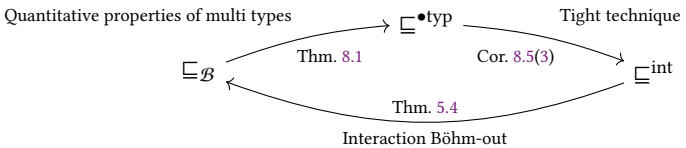
$$
\begin{array}{rrcl}
\Lambda \ni & t,u,s & ::= & x\,(\in \textsc{Var}) \mid \lambda x.t \mid tu \\
\textsc{hnfs} & h & ::= & \lambda x_1 \dots x_n.y\, t_1 \cdots t_k \\
\mathcal{C} \ni & C & ::= & \langle\cdot\rangle \mid \lambda x.C \mid t\,C \mid C\,u \\
\mathcal{H} \ni & H & ::= & \lambda x_1 \dots x_n.\langle\cdot\rangle t_1 \cdots t_k
\end{array}
\qquad
\begin{array}{rrcl}
\beta\textsc{-rule} & (\lambda x.t)u & \mapsto_\beta & t\{x := u\} \\
\eta\textsc{-rule} & \lambda x.tx & \mapsto_\eta & t, \text{if } x \notin \mathrm{fv}(t) \\
\beta & \to_\beta & := & C\langle \mapsto_\beta \rangle \\
\textsc{head} & \to_{\mathsf{h}} & := & \mathcal{H}\langle \mapsto_\beta \rangle \\
\eta & \to_\eta & := & C\langle \mapsto_\eta \rangle
\end{array}
$$

Fig. 1. The λ-calculus. Λ is the set of λ-terms, $\mathcal{C}$ the class of contexts, $\mathcal{H}$ the subclass of head contexts.

Similarly to intersection types, they *characterize* various termination properties, in the sense that t "converges" if and only if t is typable[3]. In contrast to intersection types, however, multi types are *quantitative*, which can be expressed in at least two ways. Firstly, the fact that typability implies (head) termination can be proved easily using as a decreasing measure the size of type derivations, while intersection types require more involved techniques. Secondly, multi types allow one to extract a bound of the number of head steps to normal form, as first shown by de Carvalho [2007, 2018], which is impossible with (idempotent) intersection types.

As a disclaimer, please note that multi types are a theoretical tool *not* meant to be used in real life programming languages, since typability in multi type systems is an undecidable property. Rather, they provide handy type-theoretic presentations of denotational models.

We prove that our multi type system for the checkers calculus characterizes head termination, and we use the type system to introduce a type preorder $\sqsubseteq^{\bullet\mathrm{typ}}$ and show that $\sqsubseteq_{\mathcal{B}} \subseteq \sqsubseteq^{\bullet\mathrm{typ}} \subseteq \sqsubseteq^{\mathrm{int}}$. The first of these two inclusions is proved exploiting the quantitative properties of the multi type system. The second inclusion requires more, namely to characterize the multi type judgements from which one can measure the *exact* number of interaction steps, rather than simply providing a bound. For that, we adapt the *tight* technique of Accattoli et al. [2020], in its turn refining previous work by de Carvalho [2007, 2018]. A possibly interesting point is that the literature uses the tight technique to measure the number of internal steps, while here we use it dually, for measuring interaction steps. The following diagram sums up the technical development:



In fact, we end up providing *two* characterizations of interaction equivalence/preorder, one as Böhm trees *and* one as multi types. Our first main result, namely the fact that interaction equivalence is an equational theory (Cor. 3.12(3)), is also proved using tight multi types. Essentially, it is obtained by symmetry from the inclusion $\sqsubseteq^{\bullet\mathrm{typ}} \subseteq \sqsubseteq^{\mathrm{int}}$ above and the fact that $\sqsubseteq^{\bullet\mathrm{typ}}$ is an inequational theory.

2 The λ-Calculus

To keep this article as self-contained as possible, we summarize some definitions and results concerning λ-calculus that we shall use in the paper. For more information, see [Barendregt 1984].

Definition 2.1. *The syntax and rewriting rules of λ-calculus are given in Fig. 1.*

λ-Terms and Contexts. The set Λ of *λ-terms* is constructed over a countable set Var of variables. We assume that application associates to the left and has a higher precedence than abstraction. Given $k, n \geq 0$ and $t, u, s_1, \dots, s_k \in \Lambda$, we write $\lambda\vec{x}.t$ as an abbreviation of $\lambda x_1 \dots x_n.t$, and $t\vec{s}$ for $ts_1 \cdots s_k$. For instance, $\lambda xywz.xy(wz)$ stands for $\lambda x.(\lambda y.(\lambda w.(\lambda z.((xy)(wz)))))$.

[3]The notion of convergence that is captured depends on the system that is considered. Here, we consider head normalization.

The set $\mathrm{fv}(t)$ of *free variables* of t and *α-conversion* are defined as in [Barendregt 1984, §1.2]. Hereafter, we consider λ-terms up to α-conversion and we denote α-conversion simply by $=$. The usual meta-level capture-avoiding substitution of u for x in t is noted $t\{x := u\}$.

The class $\mathcal{C}$ of *contexts* contains λ-terms built using exactly one occurrence of the *hole* $\langle\cdot\rangle$, standing for a removed sub-term. We shall also use the subclass $\mathcal{H}$ of *head contexts*. Given a context C and a λ-term t, we denote by $C\langle t\rangle$ the λ-term obtained by replacing t for the hole $\langle\cdot\rangle$ in C, possibly with capture of free variables.

Rewriting. The λ-calculus can be endowed with several *notions of reduction* $\to_{\mathsf{R}}$, turning the set Λ into a higher-order term rewriting system. Given a notion of reduction $\to_{\mathsf{R}}$:

(i) $\to_{\mathsf{R}}^*$ stands for the reflexive and transitive closure of $\to_{\mathsf{R}}$ (*multistep* R-*reduction*);
(ii) $=_{\mathsf{R}}$ stands for its reflexive, symmetric, and transitive closure (R-*conversion*);

We write $t \to_{\mathsf{R}}^n u$ to indicate a reduction sequence $t \to_{\mathsf{R}} t_1 \to_{\mathsf{R}} \cdots \to_{\mathsf{R}} t_{n-1} \to_{\mathsf{R}} u$ of length n. A λ-term t is an R-*normal form* (or R-*nf*) if there is no term u such that $t \to_{\mathsf{R}} u$. Given $t \in \Lambda$, we write $t \Downarrow_{\mathsf{R}} u$ if $t \to_{\mathsf{R}}^* u$ and u is a R-nf. The term t is R-*normalizable*, written $t \Downarrow_{\mathsf{R}}$, if $t \Downarrow_{\mathsf{R}} u$, for some u.

Contextual Closure and Reductions. We distinguish between the *(rewriting) rule* R and the *notion of reduction* $\to_{\mathsf{R}}$, where the latter is obtained from the former via some form of context closure. The context closure defined next is applicable more generally to every relation $\mathcal{R} \subseteq \Lambda^2$ on λ-terms.

Definition 2.2 (Contextual closure). *Let $\mathcal{R} \subseteq \Lambda^2$ be a relation and $\mathcal{D}$ be a class of contexts. The* $\mathcal{D}$-closure *of $\mathcal{R}$ is the least relation $\mathcal{D}\langle\mathcal{R}\rangle$ such that $t\ \mathcal{R}\ u$ entails $C\langle t\rangle\ \mathcal{D}\langle\mathcal{R}\rangle\ C\langle u\rangle$, for all $C \in \mathcal{D}$. $\mathcal{R}$ is called* $\mathcal{D}$-compatible *if $\mathcal{D}\langle\mathcal{R}\rangle \subseteq \mathcal{R}$, and simply* compatible *when $\mathcal{D}$ is the class of all contexts $\mathcal{C}$.*

The *β-reduction* $\to_\beta$ (resp. *η-reduction* $\to_\eta$) is the closure of the β-rule (η-rule) under all contexts.

Notation 2.3. Concerning specific λ-terms, we fix the following notations:

$$\mathtt{I} \coloneqq \lambda x.x, \quad \mathtt{1} \coloneqq \lambda xy.xy, \quad \mathtt{K} \coloneqq \lambda xy.x, \quad \mathtt{F} \coloneqq \lambda xy.y, \quad \mathtt{Y} \coloneqq \lambda f.(\lambda x.f(xx))(\lambda x.f(xx)), \quad \Omega \coloneqq \mathtt{YI},$$

where $\mathtt{I}$ is the identity, $\mathtt{1}$ is an η-expansion of $\mathtt{I}$, $\mathtt{K}$ and $\mathtt{F}$ are the projections, $\mathtt{Y}$ is a fixed point operator satisfying $\mathtt{Y}f =_\beta f(\mathtt{Y}f)$, and $\Omega =_\beta (\lambda x.xx)(\lambda x.xx)$ is the paradigmatic looping combinator.

Head reduction $\to_{\mathsf{h}}$, defined in Fig. 1, is the evaluation strategy adopted in this paper. We choose head reduction because of its key role in the semantics of λ-calculus [Barendregt 1984], but our construction of interaction equivalence could be adapted smoothly to weak (*i.e.* not under abstraction) head reduction. We shall discuss it in Section 10.

Fig. 1 gives the standard characterization of head normal forms (or *hnfs*). Given a hnf $h = \lambda x_1 \dots x_n.y\ t_1 \cdots t_k$, we refer to y (which may possibly be one of $x_1, \dots, x_n$) as to its *head variable*.

Conversion and Equational Theories. The equational theories of the λ-calculus, called *λ-theories*, are compatible equivalence relations containing β-reduction. They arise naturally when one aims at equating λ-terms displaying the same operational behavior. Similarly, inequational theories express the fact that the behavior of a λ-term is somewhat *less defined* than the behavior of another term.

Definition 2.4. *(i) A relation $\mathcal{R} \subseteq \Lambda^2$ is called a* congruence *if it is a compatible equivalence.*
(ii) We say that $\mathcal{R}$ is β-invariant *if it contains β-conversion $=_\beta$.*
(iii) An equational theory*, or* λ-theory*, is any β-invariant congruence $=_{\mathcal{T}}$.*
(iv) An inequational theory *is any compatible β-invariant preorder $\sqsubseteq_{\mathcal{T}}$.*
(v) An (in)equational theory is consistent *if it is different from Λ^2,* extensional *if it contains $=_\eta$.*
(vi) An inequational theory $\sqsubseteq_{\mathcal{T}}$ is semi-extensional *if it contains η-reduction $\to_\eta$.*

Note that inequational λ-theories are not required to be symmetric—they are preorders—and yet they are required to contain β-conversion $=_\beta$, which is symmetric. Any inequational λ-theory $\sqsubseteq_{\mathcal{T}}$, induces a λ-theory denoted by $=_{\mathcal{T}}$ by setting: $t =_{\mathcal{T}} u$ if both $t \sqsubseteq_{\mathcal{T}} u$ and $u \sqsubseteq_{\mathcal{T}} t$ hold.

Contextual Preorders and Equivalences. A nowadays standard notion in the study of programming languages and λ-calculi is contextual equivalence, and its asymmetric variant of contextual preorder. The idea is to observe the termination of λ-terms when plugged in the same context. Thus, they depend on a notion of termination. In this paper, we focus on termination of head reduction.

Definition 2.5. *The* (head) contextual preorder $\sqsubseteq^{\mathrm{ctx}}$ *on λ-terms is defined as follows:*

$$t \sqsubseteq^{\mathrm{ctx}} u \quad \text{if for all contexts } C\,. \quad [\, C\langle t\rangle \Downarrow_{\mathsf{h}} \quad \Rightarrow \quad C\langle u\rangle \Downarrow_{\mathsf{h}} \,]$$

The associated (head) contextual equivalence *is defined by setting $t \equiv^{\mathrm{ctx}} u$ if $t \sqsubseteq^{\mathrm{ctx}} u$ and $u \sqsubseteq^{\mathrm{ctx}} t$.*

Often the literature requires C to be a *closing* context, that is, such that both $C\langle t\rangle$ and $C\langle u\rangle$ are closed. In the ordinary λ-calculus, adding/removing the closing requirement does not change the defined relation, which is why we do not add it. It turns out that the head contextual preorder provides an inequational λ-theory.

Theorem 2.6 ([Barendregt 1984]). *The head contextual preorder $\sqsubseteq^{\mathrm{ctx}}$ is an inequational λ-theory. Moreover, it is consistent and extensional.*

As mentioned in the introduction, the proof of the previous theorem is non-trivial. Proving that β-conversion $=_\beta$ is included in $\sqsubseteq^{\mathrm{ctx}}$ indeed is not immediate, because of the famously fastidious universal quantification on contexts in $\sqsubseteq^{\mathrm{ctx}}$. A rewriting-based proof requires the use of both confluence of β and the untyped normalization theorem of head reduction (if $t \to_\beta^* u$ and u is normal for head reduction, then t is head normalizing). For an overview, see Appendix A of the longer version on arXiv [Accattoli et al. 2024]. Similarly, a rewriting-based proof that η-conversion $=_\eta$ is included in $\sqsubseteq^{\mathrm{ctx}}$ also rests on non-trivial theorems about η. Also in this case, for an overview see Appendix A of the longer version on arXiv [Accattoli et al. 2024].

Semantic proofs are possible but not easy anyway, as they rest on soundness of the model. For the similar case of interaction equivalence, we shall develop a semantic proof.

Background and Notable Variants. The head contextual preorder has been studied in-depth because it captures the (in)equational theory of Scott's denotational model $\mathcal{D}_\infty$ [Scott 1972], the first model of the untyped λ-calculus. The preorder $\sqsubseteq^{\mathrm{ctx}}$ is also the maximal contextual preorder of interest: any strictly larger inequational λ-theory is inconsistent [Barendregt and Manzonetto 2022, Lemma 12.5]. In 1976, Hyland and Wadsworth have shown that the associated equivalence $\equiv^{\mathrm{ctx}}$ coincides with Böhm trees equality up to infinite η-expansions (Cf. Theorem 4.7, below).

Another relevant contextual preorder is obtained by considering termination with respect to β-reduction, instead of head reduction. This alternative preorder was originally introduced in Morris's PhD thesis [1968], but has been the subject of fewer investigations than the head one (until recently, see [Intrigila et al. 2019] and [Barendregt and Manzonetto 2022, Ch. 12]). Hyland [1975] showed that the associated equivalence captures Böhm trees equality up to *finite* η-expansions.

3 The Checkers Calculus

In this section we introduce the *checkers calculus*, obtained from the λ-calculus by duplicating the abstraction and application constructors, that now both come in white and black dresses.

Definition 3.1. *The syntax and operational semantics of the checkers calculus are defined in Fig. 2.*

Black and White Constructors. The constructors of abstraction and application receive a tag, called *player*, that can be either white $\circ$ or black $\bullet$. The terms populating the set $\Lambda_{\circ\bullet}$ are called *checkers terms*, and inherit the notions of free variables, α-conversion, and substitution from λ-calculus. Note that variables do *not* receive a player tag. The main reason is simplicity: this way, we do not need to enforce the uniform tagging of all the occurrences of the same variable.

Terms and Contexts	Beta rules and reductions
Players $\quad \mathsf{p}, \mathsf{q} \quad ::= \quad \circ \mid \bullet$	Silent $\quad (\lambda_{\mathsf{p}} x.t) \cdot^{\mathsf{p}} u \quad \mapsto_{\beta_\tau} \quad t\{x := u\}$
Terms $\Lambda_{\circ\bullet} \ni \quad t, u, s \quad ::= \quad x \mid \lambda_{\mathsf{p}} x.t \mid t \cdot^{\mathsf{p}} u$	Interaction $\quad (\lambda_{\mathsf{p}} x.t) \cdot^{\mathsf{p}^\perp} u \quad \mapsto_{\beta_{\textcircled{\bullet}}} \quad t\{x := u\}$
$\mathsf{h}_{\circ\bullet}$-nfs $\quad h, h' \quad ::= \quad \lambda_{\vec{\mathsf{p}}} \vec{x}.y \cdot^{\mathsf{q}_1 \cdots \mathsf{q}_k} t_1 \cdots t_k$	Silent $\beta \quad \to_{\beta_\tau} \quad := \quad C_{\circ\bullet}\langle \mapsto_{\beta_\tau} \rangle$
	Interaction $\beta \quad \to_{\beta_{\textcircled{\bullet}}} \quad := \quad C_{\circ\bullet}\langle \mapsto_{\beta_{\textcircled{\bullet}}} \rangle$
$C_{\circ\bullet} \ni \quad C \quad ::= \quad \langle\cdot\rangle \mid \lambda_{\mathsf{p}} x.C \mid C \cdot^{\mathsf{p}} u \mid t \cdot^{\mathsf{p}} C$	Checkers $\beta \quad \to_{\beta_{\circ\bullet}} \quad := \quad \to_{\beta_\tau} \cup \to_{\beta_{\textcircled{\bullet}}}$
$\mathcal{H}_{\circ\bullet} \ni \quad H \quad ::= \quad \lambda_{\vec{\mathsf{p}}} \vec{x}.\langle\cdot\rangle \cdot^{\mathsf{q}_1 \cdots \mathsf{q}_k} t_1 \cdots t_k$	Silent head $\quad \to_{\mathsf{h}_\tau} \quad := \quad \mathcal{H}_{\circ\bullet}\langle \mapsto_{\beta_\tau} \rangle$
	Interaction head $\quad \to_{\mathsf{h}_{\textcircled{\bullet}}} \quad := \quad \mathcal{H}_{\circ\bullet}\langle \mapsto_{\beta_{\textcircled{\bullet}}} \rangle$
	Checkers head $\quad \to_{\mathsf{h}_{\circ\bullet}} \quad := \quad \to_{\mathsf{h}_\tau} \cup \to_{\mathsf{h}_{\textcircled{\bullet}}}$

Fig. 2. The checkers calculus.

Notation 3.2. When the players are actually specified, we simply denote the applications by $t \circ u$ (and $t \bullet u$) instead of $t \cdot^{\circ} u$ (and $t \cdot^{\bullet} u$). Hereafter, we often need to refer to constructors of a fixed but arbitrary player p, thus we use $\lambda_{\mathsf{p}} x.t$ and $t \cdot^{\mathsf{p}} u$ for abstractions and applications of player $\mathsf{p} \in \{\circ, \bullet\}$. We also use $\mathsf{p}^\perp$ to denote the opposite player, defined as $\circ^\perp := \bullet$ and $\bullet^\perp := \circ$. In case of many consecutive abstractions or applications, we shorten the notations to $\lambda_{\vec{\mathsf{p}}} \vec{x}.t$ and $t \cdot^{\vec{\mathsf{p}}} \vec{u}$, respectively. The former is sometimes slightly expanded to $\lambda_{\mathsf{p}_1 \cdots \mathsf{p}_k} x_1 \ldots x_k.\, t$, and the latter to $t \cdot^{\mathsf{p}_1 \cdots \mathsf{p}_k} u_1 \cdots u_k$.

Checkers Contexts. The class $C_{\circ\bullet}$ of *checkers contexts*, and the subclass $\mathcal{H}_{\circ\bullet}$ of *checkers head contexts* are defined in Fig. 2. Definition 2.2 of contextual closure with respect to a class of contexts, generalizes to this setting in the obvious way. Checkers contexts shall play a key role in the definition of contextual preorders and equivalences on checkers terms (see Definition 3.7, below).

Silent and Interaction Steps. There are two kinds of colored β-redexes $(\lambda_{\mathsf{p}} x.t) \cdot^{\mathsf{q}} u$, the silent one and the interaction one, each one with its own β-rule. Silent redexes are β-redexes where the color of the abstraction p matches the color of the application q. Intuitively, these steps are internal to each player's world. In interaction redexes, instead, the color of the abstraction and the color of the application are different, *i.e.* $\mathsf{p} \neq \mathsf{q}$. This represents the scenario where the two players interact with each other, which, from each player's perspective, amounts to interacting with the external world. Our focus shall be on the number of head interaction steps.

The internal/external dichotomy, and the idea of having two players are strong guiding intuitions but note that, in general, a checkers term can arbitrarily interleave black and white constructors: there is no neat frontier between the parts of a term corresponding to the two players. The key point is that, even if we start with cleanly separated black and white parts, they still end up mixing during the reduction.

Example 3.3. *Consider the black identity* $\mathtt{I}_\bullet := \lambda_\bullet x.x$ *and the white term* $\mathtt{D}_\circ := \lambda_\circ y.\lambda_\circ x.x \circ (y \circ x)$.

(1) If the black identity is black-applied to $\mathtt{D}_\circ$*, then it gives rise to a silent step* $\mathtt{I}_\bullet \bullet \mathtt{D}_\circ \to_{\beta_\tau} \mathtt{D}_\circ$*, while if it is white-applied to* $\mathtt{D}_\circ$ *then the step is an interaction one, namely* $\mathtt{I}_\bullet \circ \mathtt{D}_\circ \to_{\beta_{\textcircled{\bullet}}} \mathtt{D}_\circ$.

(2) $\mathtt{D}_\circ \bullet \mathtt{I}_\bullet \bullet \mathtt{I}_\bullet \to_{\mathsf{h}_{\textcircled{\bullet}}} (\lambda_\circ x.x \circ (\mathtt{I}_\bullet \circ x)) \bullet \mathtt{I}_\bullet \to_{\beta_{\textcircled{\bullet}}} (\lambda_\circ x.x \circ x) \bullet \mathtt{I}_\bullet \to_{\mathsf{h}_{\textcircled{\bullet}}} \mathtt{I}_\bullet \circ \mathtt{I}_\bullet \to_{\mathsf{h}_{\textcircled{\bullet}}} \mathtt{I}_\bullet$

Basic Rewriting Properties. As the λ-calculus, the checkers λ-calculus is an example of *orthogonal higher-order rewriting system* [Aczel 1978; Klop 1980; Nipkow 1991], that is a class of rewriting systems for which confluence always holds, because of the good shape of their rewriting rules. Similarly, the silent and interaction sub-relations $\to_{\beta_\tau}$ and $\to_{\beta_{\textcircled{\bullet}}}$ are also confluent and commute.

Theorem 3.4 (Confluence). *Reductions* $\to_{\beta_{\circ\bullet}}$, $\to_{\beta_\tau}$, *and* $\to_{\beta_{\textcircled{\bullet}}}$ *are confluent.*

Similarly, checkers head reduction inherits various expected properties of head reduction, such as determinism. In particular, we shall use the following immediate substitutivity property.

Lemma 3.5 (Substitutivity). *Let $t, t', u \in \Lambda_{\circ\bullet}$ and $\mathsf{R} \in \{\beta_\tau, \beta_{\bullet\bullet}, \beta_{\circ\bullet}, \mathsf{h}_\tau, \mathsf{h}_{\bullet\bullet}, \mathsf{h}_{\circ\bullet}\}$. If $t \to_{\mathsf{R}} t'$ then $t\{x := u\} \to_{\mathsf{R}} t'\{x := u\}$.*

η-Conversion. Note that there are no checkers η-rules in Fig. 2. A key point in our work, indeed, is that η-conversion can change the number of interaction steps.

Example 3.6 (The Delicate Role of η). *Consider the black η-expansion $\mathtt{1}_\bullet := \lambda_\bullet x.\lambda_\bullet y.x \bullet y$ of the black identity $\mathtt{I}_\bullet$ and the following diagram:*

$$\mathtt{1}_\bullet \circ z \circ w \;\begin{array}{l} \xrightarrow{\mathsf{h}_{\bullet\bullet}} (\lambda_\bullet y.z \bullet y) \circ w \xrightarrow{\mathsf{h}_{\bullet\bullet}} z \bullet w \\ \xrightarrow{\bullet\eta} \mathtt{I}_\bullet \circ z \circ w \xrightarrow{\mathsf{h}_{\bullet\bullet}} z \circ w \end{array} \qquad \neq$$

It shows that η changes the number of interaction steps. It also shows that standard properties of ordinary η do not lift to the checkers case (see Appendix A on arXiv [Accattoli et al. 2024] for definitions): $\to_\eta$ cannot be postponed after $\to_{\mathsf{h}_{\bullet\bullet}}$, $\to_\eta$ and $\to_{\mathsf{h}_{\bullet\bullet}}$ do not commute, and adding η to $\to_{\beta_{\circ\bullet}}$ breaks confluence.

Big-Step Notation and Interaction Index. Recall that $t \Downarrow_{\mathsf{h}_{\circ\bullet}}$ stands for "t is $\mathsf{h}_{\circ\bullet}$-normalizing". We introduce the notation $t \Downarrow^{\bullet\bullet k}_{\mathsf{h}_{\circ\bullet}}$ for: t is $\mathsf{h}_{\circ\bullet}$-normalizing *and* the number of head interaction steps $\to_{\mathsf{h}_{\bullet\bullet}}$ in its $\mathsf{h}_{\circ\bullet}$-evaluation is k. Note that the number k is well-defined because $\to_{\mathsf{h}_{\circ\bullet}}$ is deterministic.

Interaction Equivalence and Preorders. We now introduce interaction equivalence $\equiv^{\mathrm{ctx}}_{\bullet\bullet}$ as a form of quantitative contextual equivalence for the checkers calculus. Similarly to weak similarity for labeled transition systems, $\equiv^{\mathrm{ctx}}_{\bullet\bullet}$ ignores silent head steps. The quantitative aspect is that it requires to preserve the number of interaction head steps. The richer quantitative setting in fact gives rise to *two* possible preorders, both generating $\equiv^{\mathrm{ctx}}_{\bullet\bullet}$ when symmetrized.

Definition 3.7 (Checkers interaction preorders and equivalence). *We define the* interaction preorder $\sqsubseteq^{\mathrm{ctx}}_{\bullet\bullet}$, *the* interaction improvement (preorder) $\sqsubseteq^{\mathrm{ctx\cdot imp}}_{\bullet\bullet}$, *and the* interaction equivalence $\equiv^{\mathrm{ctx}}_{\bullet\bullet}$ *on checkers terms $t, u \in \Lambda_{\circ\bullet}$ as follows:*

(i) $t \sqsubseteq^{\mathrm{ctx}}_{\bullet\bullet} u$ *if* $C\langle t\rangle \Downarrow^{\bullet\bullet k}_{\mathsf{h}_{\circ\bullet}}$ *implies* $C\langle u\rangle \Downarrow^{\bullet\bullet k}_{\mathsf{h}_{\circ\bullet}}$ *for all checkers contexts $C \in \mathcal{C}_{\circ\bullet}$ and $k \in \mathbb{N}$;*
(ii) $t \sqsubseteq^{\mathrm{ctx\cdot imp}}_{\bullet\bullet} u$ *if* $C\langle t\rangle \Downarrow^{\bullet\bullet k}_{\mathsf{h}_{\circ\bullet}}$ *implies* $C\langle u\rangle \Downarrow^{\bullet\bullet k'}_{\mathsf{h}_{\circ\bullet}}$ *with $k' \leq k$, for all contexts $C \in \mathcal{C}_{\circ\bullet}$ and $k \in \mathbb{N}$;*
(iii) $t \equiv^{\mathrm{ctx}}_{\bullet\bullet} u$ *is the equivalence relation induced by $\sqsubseteq^{\mathrm{ctx}}_{\bullet\bullet}$, that is, $t \equiv^{\mathrm{ctx}}_{\bullet\bullet} u$ if $t \sqsubseteq^{\mathrm{ctx}}_{\bullet\bullet} u$ and $u \sqsubseteq^{\mathrm{ctx}}_{\bullet\bullet} t$.*

The interaction improvement $\sqsubseteq^{\mathrm{ctx\cdot imp}}_{\bullet\bullet}$ adapts Sands' improvements [1996a; 1996b; 1999] to our *up to silent steps* setting. Note that $\sqsubseteq^{\mathrm{ctx}}_{\bullet\bullet} \subseteq \sqsubseteq^{\mathrm{ctx\cdot imp}}_{\bullet\bullet}$. It is easy to see that the two new preorders are different, *i.e.* that $\sqsubseteq^{\mathrm{ctx}}_{\bullet\bullet} \subsetneq \sqsubseteq^{\mathrm{ctx\cdot imp}}_{\bullet\bullet}$. Indeed, $\mathtt{I}_\bullet \circ \mathtt{I}_\circ \sqsubseteq^{\mathrm{ctx\cdot imp}}_{\bullet\bullet} \mathtt{I}_\circ$ but $\mathtt{I}_\bullet \circ \mathtt{I}_\circ \not\sqsubseteq^{\mathrm{ctx}}_{\bullet\bullet} \mathtt{I}_\circ$, as both checkers terms have $\mathtt{I}_\circ$ as head normal form but $\mathtt{I}_\bullet \circ \mathtt{I}_\circ$ requires one more interaction step to reach it.

The interaction preorder $\sqsubseteq^{\mathrm{ctx}}_{\bullet\bullet}$ turns out to be more easily manageable than the interaction improvement $\sqsubseteq^{\mathrm{ctx\cdot imp}}_{\bullet\bullet}$. Moreover, from the inclusion $\sqsubseteq^{\mathrm{ctx}}_{\bullet\bullet} \subseteq \sqsubseteq^{\mathrm{ctx\cdot imp}}_{\bullet\bullet}$ we shall be able to transfer some properties of $\sqsubseteq^{\mathrm{ctx}}_{\bullet\bullet}$ to $\sqsubseteq^{\mathrm{ctx\cdot imp}}_{\bullet\bullet}$. In this paper, then, we shall rather focus on $\sqsubseteq^{\mathrm{ctx}}_{\bullet\bullet}$.

Example 3.8. *Recall that $\mathtt{I}_\bullet$ and $\mathtt{D}_\circ$ have been defined in Ex. 3.3, and $\mathtt{1}_\bullet$ in Ex. 3.6.*

(i) $(\lambda_\circ x.x \circ x) \circ (\lambda_\circ x.x \circ x) \sqsubseteq^{\mathrm{ctx}}_{\bullet\bullet} \mathtt{I}_\bullet$, *because the former is not $\mathsf{h}_{\circ\bullet}$-normalizable.*
(ii) $\mathtt{I}_\bullet \not\sqsubseteq^{\mathrm{ctx}}_{\bullet\bullet} \mathtt{1}_\bullet$ *and* $\mathtt{1}_\bullet \not\sqsubseteq^{\mathrm{ctx}}_{\bullet\bullet} \mathtt{I}_\bullet$, *as they are separated by $C = \langle\cdot\rangle \circ z \circ w$, see Ex. 3.6.*
(iii) $\mathtt{D}_\circ \circ (\lambda x_\circ.x) \circ (\lambda x_\circ.x) \equiv^{\mathrm{ctx}}_{\bullet\bullet} (\lambda x_\circ.x)$, *because they are β_τ-convertible.*

Back to Ordinary λ-Terms. Composing the interaction relations for checkers terms defined above with the *black embedding* of Λ into $\Lambda_{\circ\bullet}$, we obtain new *interaction* relations on ordinary λ-terms.

Definition 3.9 (Player Lifting and interaction preorders/equivalence).

(i) Given $\mathrm{p} \in \{\circ, \bullet\}$*, define the* player p-lifting of ordinary λ-terms and ordinary contexts*, as the maps* $\overline{\cdot}^{\mathrm{p}} : \Lambda \to \Lambda_{\circ\bullet}$ *and* $\overline{\cdot}^{\mathrm{p}} : C \to C_{\circ\bullet}$ *obtained by* p*-tagging every constructor:*

$$\overline{x}^{\mathrm{p}} := x, \qquad \overline{\lambda x.t}^{\mathrm{p}} := \lambda_{\mathrm{p}} x.\overline{t}^{\mathrm{p}}, \qquad \overline{tu}^{\mathrm{p}} := \overline{t}^{\mathrm{p}} \cdot^{\mathrm{p}} \overline{u}^{\mathrm{p}};$$
$$\overline{\langle\cdot\rangle}^{\mathrm{p}} := \langle\cdot\rangle, \qquad \overline{\lambda x.C}^{\mathrm{p}} := \lambda_{\mathrm{p}} x.\overline{C}^{\mathrm{p}}, \qquad \overline{tC}^{\mathrm{p}} := \overline{t}^{\mathrm{p}} \cdot^{\mathrm{p}} \overline{C}^{\mathrm{p}}, \qquad \overline{Cu}^{\mathrm{p}} := \overline{C}^{\mathrm{p}} \cdot^{\mathrm{p}} \overline{u}^{\mathrm{p}}.$$

(ii) The interaction preorder $\sqsubseteq^{\mathrm{int}}$*, the* interaction improvement $\sqsubseteq^{\mathrm{int}\cdot\mathrm{imp}}$*, and the* interaction equivalence $\equiv^{\mathrm{int}}$ *are the following relations on ordinary* λ*-terms* $t, u \in \Lambda$ *defined via black lifting:*

- $t \sqsubseteq^{\mathrm{int}} u$ *if* $\overline{t}^{\bullet} \sqsubseteq^{\mathrm{ctx}}_{\bullet} \overline{u}^{\bullet}$*;*
- $t \sqsubseteq^{\mathrm{int}\cdot\mathrm{imp}} t'$ *if* $\overline{t}^{\bullet} \sqsubseteq^{\mathrm{ctx}\cdot\mathrm{imp}}_{\bullet} \overline{u}^{\bullet}$*;*
- $t \equiv^{\mathrm{int}} u$ *is the equivalence relation induced by* $\sqsubseteq^{\mathrm{int}}$*, that is,* $t \equiv^{\mathrm{int}} u$ *if* $t \sqsubseteq^{\mathrm{int}} u$ *and* $u \sqsubseteq^{\mathrm{int}} t$*.*

Example 3.10. *We use the* λ*-terms introduced in Notation 2.3. We also consider* $\mathtt{D} := \lambda yx.x(yx)$*, having the property that* $\mathtt{YD}$ *is a fixed point operator whose head reduction is 'slower' than that of* $\mathtt{Y}$*.*

(i) $\mathtt{YK} \sqsubseteq^{\mathrm{int}} t$*, for all* $t \in \Lambda$*, because* $\mathtt{YK}$ *does not have an hnf. Similarly,* $\mathtt{YI} =_{\beta} \Omega \sqsubseteq^{\mathrm{int}} t$*.*

(ii) We have $\mathtt{I} \not\sqsubseteq^{\mathrm{int}} \mathtt{1}$*, nor* $\mathtt{1} \not\sqsubseteq^{\mathrm{int}} \mathtt{I}$ *(Cf. Ex. 3.8(ii)). It is easily seen that* $\mathtt{K} \not\sqsubseteq^{\mathrm{int}} \mathtt{F}$ *and* $\mathtt{F} \not\sqsubseteq^{\mathrm{int}} \mathtt{K}$*.*

(iii) $\mathtt{YD} \equiv^{\mathrm{int}} \mathtt{Y}$*, as the former needs more head-reduction steps to converge, but they are silent.*

The Interaction Preorder is Inequational. We now show that the interaction preorder $\sqsubseteq^{\mathrm{int}}$ is a semantics of λ-calculus, *i.e.*, it is an inequational λ-theory, from which the corresponding results for $\sqsubseteq^{\mathrm{int}\cdot\mathrm{imp}}$ and $\equiv^{\mathrm{int}}$ follow. This is our first main result, showing that our framework does solve the internal/external tension evoked in the introduction.

As for contextual equivalence, proving invariance (that is, that β-conversion is included) is non-trivial, and even harder because of the constraint on the number of interaction steps. It follows from the following theorem for the checkers interaction preorder, which we shall prove in a later section via a semantic proof based on multi types (page 23), rather than via rewriting-based techniques.

Theorem 3.11 (The interaction preorder includes silent conversion). *For all checkers terms* $t, u \in \Lambda_{\circ\bullet}$*,* $t =_{\beta_\tau} u$ *entails* $t \sqsubseteq^{\mathrm{ctx}}_{\bullet} u$*.*

Corollary 3.12.

(1) The interactional preorder $\sqsubseteq^{\mathrm{int}}$ *is a consistent inequational* λ*-theory. Moreover, it is not semi-extensional, whence not extensional.*

(2) The interactional improvement $\sqsubseteq^{\mathrm{int}\cdot\mathrm{imp}}$ *is a consistent inequational* λ*-theory.*

(3) Interaction equivalence $\equiv^{\mathrm{int}}$ *is a consistent* λ*-theory. Moreover, it is not extensional.*

Proof. (3) follows from (1).

(1) We unfold the definition of inequational λ-theory, and check the following properties.
- *Preorder*. Reflexivity and transitivity are straightforward.
- *Compatibility*. It follows from the compatibility of $\sqsubseteq^{\mathrm{ctx}}_{\bullet}$. If $t \sqsubseteq^{\mathrm{int}} u$, then we need to prove that $C\langle t\rangle \sqsubseteq^{\mathrm{int}} C\langle u\rangle$, for any context C. Let C' be a context. If $\overline{C'\langle C\langle t\rangle\rangle}^{\bullet} \Downarrow^{\bullet k}_{\mathsf{h}_{\circ\bullet}}$ then $\overline{C'\langle C\langle u\rangle\rangle}^{\bullet} \Downarrow^{\bullet k}_{\mathsf{h}_{\circ\bullet}}$ by $\overline{t}^{\bullet} \sqsubseteq^{\mathrm{ctx}}_{\bullet} \overline{u}^{\bullet}$.
- *Invariance*. Let $t =_{\beta} u$. Then clearly $\overline{t}^{\bullet} =_{\beta_\tau} \overline{u}^{\bullet}$. By Thm. 3.11, $\overline{t}^{\bullet} \sqsubseteq^{\mathrm{ctx}}_{\bullet} \overline{u}^{\bullet}$. Then $t \sqsubseteq^{\mathrm{int}} u$.

  Consistency of $\sqsubseteq^{\mathrm{int}}$ is given by the fact that $\mathtt{I} \not\sqsubseteq^{\mathrm{int}} \Omega$, as it can be seen by considering the empty context. The failure of semi-extensionality is shown in Ex. 3.6.

(2) The proof that $\sqsubseteq^{\mathrm{int}\cdot\mathrm{imp}}$ is a compatible and consistent goes as for $\sqsubseteq^{\mathrm{int}}$. For invariance, just note that $=_{\beta} \subseteq \sqsubseteq^{\mathrm{int}} \subseteq \sqsubseteq^{\mathrm{int}\cdot\mathrm{imp}}$. □

Interaction Improvement and η. Note that the corollary does not say anything about $\sqsubseteq^{\mathrm{int}\cdot\mathrm{imp}}$ and η. We conjecture that η-reduction is included in $\sqsubseteq^{\mathrm{int}\cdot\mathrm{imp}}$, which would imply that the obvious inclusion $\sqsubseteq^{\mathrm{int}} \subseteq \sqsubseteq^{\mathrm{int}\cdot\mathrm{imp}}$ is strict, since it is not included in $\sqsubseteq^{\mathrm{int}}$ (Ex. 3.6) (beware that strictness of the inclusion does not follows from $\sqsubseteq^{\mathrm{ctx}}_{\bullet} \subsetneq \sqsubseteq^{\mathrm{ctx}\cdot\mathrm{imp}}_{\bullet}$ as strictness in that case relies on black *and* white terms). For instance, $1 \to_\eta \mathtt{I}$ and we expect that $1 \sqsubseteq^{\mathrm{int}\cdot\mathrm{imp}} \mathtt{I}$ because, intuitively, head termination of $C\langle 1_\bullet\rangle$ does require at most one more interaction step than for $C\langle \mathtt{I}_\bullet\rangle$, for all $C \in \mathcal{C}_{\circ\bullet}$.

At present, however, it is only a conjecture. There is a real technical difficulty because the properties of η that are usually used for rewriting-based proofs of similar facts fail for the checkers calculus, see Ex. 3.6. The semantic tools that we shall develop in the next sections for studying $\sqsubseteq^{\mathrm{int}}$ are not able to deal with η either.

Adding and Removing Players. The next proposition formalizes the fact that the black-only and white-only fragments of the checkers calculus are embeddings of the ordinary λ-calculus, not only statically but also dynamically. The statements concern head steps, but generalize to arbitrary steps.

Proposition 3.13 (Lifting Properties). *Let $\mathsf{p} \in \{\circ, \bullet\}$ and $t, u \in \Lambda$:*

(1) Head steps are turned into silent ones*: if $t \to_{\mathsf{h}} u$ then $\overline{t}^{\mathsf{p}} \to_{\mathsf{h}_\tau} \overline{u}^{\mathsf{p}}$.*
(2) Mono-player head-steps can be pulled back to the ordinary λ-calculus*: if $\overline{t}^{\mathsf{p}} \to_{\mathsf{h}\circ\bullet} s$ then there exists a λ-term $u \in \Lambda$ such that $t \to_{\mathsf{h}} u$ and $s = \overline{u}^{\mathsf{p}}$.*
(3) Head normal forms are preserved*: t is a hnf if and only if $\overline{t}^{\mathsf{p}}$ is $\mathsf{h}_{\circ\bullet}$-normal.*

Let us also formalize the fact that adding tags to application and abstractions does not change the possible reductions. Let $t \in \Lambda$, a tagging t^T is a checkers term that preserves the syntax but tags its abstractions and applications with players.

Proposition 3.14. *Let $t \to_{\mathsf{h}} u$ be a reduction in the ordinary λ-calculus. For any tagging T of t, there exists a tagging T' of u such that $t^T \to_{\mathsf{h}\circ\bullet} u^{T'}$ in the checkers calculus.*

It follows that sequences of head reductions in the ordinary λ-calculus are preserved in the checkers calculus, under some tagging determined by the first term in the reduction.

Hierarchy. Since $\sqsubseteq^{\mathrm{ctx}}_{\bullet} \subseteq \sqsubseteq^{\mathrm{ctx}\cdot\mathrm{imp}}_{\bullet}$, we have $\sqsubseteq^{\mathrm{int}} \subseteq \sqsubseteq^{\mathrm{int}\cdot\mathrm{imp}}$ (we showed that $\sqsubseteq^{\mathrm{ctx}}_{\bullet} \subsetneq \sqsubseteq^{\mathrm{ctx}\cdot\mathrm{imp}}_{\bullet}$, but not by using black lifted terms, so that does not give us $\sqsubseteq^{\mathrm{int}} \subsetneq \sqsubseteq^{\mathrm{int}\cdot\mathrm{imp}}$). The lifting properties above are used to prove the following expected lemma.

Lemma 3.15. $\sqsubseteq^{\mathrm{int}\cdot\mathrm{imp}} \subseteq \sqsubseteq^{\mathrm{ctx}}$.

From Ex. 3.6, it follows that $\mathtt{I} \not\sqsubseteq^{\mathrm{int}\cdot\mathrm{imp}} 1$, while $\mathtt{I} \sqsubseteq^{\mathrm{ctx}} 1$ holds. Thus, $\sqsubseteq^{\mathrm{int}\cdot\mathrm{imp}} \subsetneq \sqsubseteq^{\mathrm{ctx}}$. Summing up, we obtain the following hierarchy of preorders.

Lemma 3.16 (Hierarchy). $\sqsubseteq^{\mathrm{int}} \subseteq \sqsubseteq^{\mathrm{int}\cdot\mathrm{imp}} \subsetneq \sqsubseteq^{\mathrm{ctx}}$. *Similarly,* $\equiv^{\mathrm{int}} \subsetneq \equiv^{\mathrm{ctx}}$.

4 Böhm Trees

Here starts the second part of the paper, where interaction equivalence $\equiv^{\mathrm{int}}$ shall be characterized as the equational theory $\mathcal{B}$ induced by the equality of Böhm trees. In this section, we recall Böhm trees and two notions of equality between them.

Barendregt proposed to represent the (possibly infinite) behavior of a λ-term t as a (possibly infinite) tree, obtained by repeatedly slicing it with respect to head termination. We first present the idea informally, and then formally, following the similarity-based approach of Lassen [1999].

Definition 4.1 (Barendregt 1977). *The* Böhm tree $\mathsf{BT}(t)$ *of a λ-term t is defined as follows:*

- *If t is head terminating then $t \to_{\mathsf{h}}^* \lambda x_1 \dots x_n.y\, t_1 \cdots t_k$, for some $n, k \geq 0$, and we define:*

$$\mathsf{BT}(t) \coloneqq \quad \lambda x_1 \dots x_n.y \quad \text{with children } \mathsf{BT}(t_1) \cdots \mathsf{BT}(t_k)$$

- *Otherwise, t is head diverging and we define* $\mathsf{BT}(t) \coloneqq \bot$.

Böhm trees are naturally ordered as follows: $\mathsf{BT}(t) \leq_\bot \mathsf{BT}(u)$ *whenever* $\mathsf{BT}(u)$ *is obtained from* $\mathsf{BT}(t)$ *by replacing some (possibly zero, or infinitely many) occurrences of* $\bot$ *by arbitrary trees.*

Intuitively, the constant $\bot$ represents the complete lack of information and, accordingly, the relation $\mathsf{BT}(t) \leq_\bot \mathsf{BT}(u)$ captures the fact that the behavior of u is more defined than that of t.

Example 4.2. *Some examples of Böhm trees of notable λ-terms:*

$$\mathsf{BT}(\Omega) = \bot \qquad \mathsf{BT}(\mathsf{P}z) = \lambda x_0.x_0,\ \lambda x_1.x_1,\ \lambda x_2.x_2,\dots \qquad \mathsf{BT}(\mathsf{Y}) = \lambda f.f,\ f,\ f,\dots \quad \geq_\bot \quad \lambda f.f,\ f,\ \bot \quad \geq_\bot \quad \lambda f.f,\ \bot \quad \geq_\bot \quad \bot$$

$$\mathsf{BT}(\mathsf{1}) = \lambda xy.x \text{ with child } y$$

where $\mathsf{P} = \mathsf{Y}(\lambda yzx.x(yz))$ *satisfies* $\mathsf{P}z =_\beta \lambda x.x(\mathsf{P}z)$. *Note that z is never erased by* P*, rather "pushed into infinity" in the sense that* $\mathsf{P}z \to_\beta^* t$ *entails* $z \in \mathsf{fv}(t)$*, but z does not occur in* $\mathsf{BT}(\mathsf{P}z)$.

It is well-known that Böhm trees are invariant under β-conversion [Barendregt 1984, Ch. 10]. Informally, the Böhm preorder $\sqsubseteq_{\mathcal{B}}$ on terms is defined by pulling back the preorder on the associated trees, that is, $t \sqsubseteq_{\mathcal{B}} u$ if $\mathsf{BT}(t) \leq_\bot \mathsf{BT}(u)$. Formally, we rather define the Böhm preorder as a notion of similarity, following Lassen [1999].

Definition 4.3 (Böhm preorder, formally). *The Böhm preorder $\sqsubseteq_{\mathcal{B}}$, also known as* head (normal form) similarity*, is the largest relation $t \sqsubseteq_{\mathcal{B}} u$ closed under the following clauses:*

(bot) $t \not\Downarrow_{\mathrm{h}}$ *i.e. t has no head normal form.*
(H) $t \Downarrow_{\mathrm{h}} \lambda x_1 \dots x_n.y\, t_1 \cdots t_k$ *and* $u \Downarrow_{\mathrm{h}} \lambda x_1 \dots x_n.y\, u_1 \cdots u_k$ *with* $(t_i \sqsubseteq_{\mathcal{B}} u_i)_{i \leq k}$.

Theorem 4.4 ([Barendregt 1984, Cor. 14.3.20(iii)]). *The Böhm preorder $\sqsubseteq_{\mathcal{B}}$ is an inequational λ-theory. Moreover, $\sqsubseteq_{\mathcal{B}}$ is neither extensional nor semi-extensional.*

The main result of the paper is that $\sqsubseteq_{\mathcal{B}}$ coincides with the interaction preorder $\sqsubseteq^{\mathrm{int}}$. For the direction $\sqsubseteq^{\mathrm{int}} \subseteq \sqsubseteq_{\mathcal{B}}$, we shall partly rely on a similar important result in the theory of the λ-calculus, namely Hyland's semi-separation theorem, which concerns an extensional variant of $\sqsubseteq_{\mathcal{B}}$ below.

Extensional Böhm Preorder. We now formally introduce an extensional version of $\sqsubseteq_{\mathcal{B}}$ capturing the preorder induced by Scott's model $\mathcal{D}_\infty$.

Definition 4.5 (Extensional Böhm preorder). *The* extensional Böhm preorder $\sqsubseteq_{\mathcal{B}\eta^\infty}$ *is the largest relation $t \sqsubseteq_{\mathcal{B}\eta^\infty} u$ closed under the following clauses:*

(bot) $t \not\Downarrow_{\mathrm{h}}$ *i.e. t has no head normal form.*
(Hη) $t \Downarrow_{\mathrm{h}} h$, $u \Downarrow_{\mathrm{h}} h'$, *and there exist* $n, k \geq 0$ *such that* $h =_\eta \lambda x_1 \dots x_n.y\, t_1 \cdots t_k$ *and* $h' =_\eta \lambda x_1 \dots x_n.y\, u_1 \cdots u_k$ *with* $(t_i \sqsubseteq_{\mathcal{B}\eta^\infty} u_i)_{i \leq k}$.

Example 4.6. *Recall that* I, $\mathsf{1}$, K, *and* F *have been defined in Notation 2.3.*

(i) We start with some negative examples: $\mathsf{K} \not\sqsubseteq_{\mathcal{B}\eta^\infty} \mathsf{F}$ *and* $\mathsf{F} \not\sqsubseteq_{\mathcal{B}\eta^\infty} \mathsf{K}$ *(their head variables differ).*
(ii) Both $\mathsf{I} \sqsubseteq_{\mathcal{B}\eta^\infty} \mathsf{1}$ *and* $\mathsf{1} \sqsubseteq_{\mathcal{B}\eta^\infty} \mathsf{I}$ *hold. This entails the extensionality of* $\sqsubseteq_{\mathcal{B}\eta^\infty}$*, hence of* $=_{\mathcal{B}\eta^\infty}$.
(iii) $\lambda xy.x\Omega \sqsubseteq_{\mathcal{B}\eta^\infty} \mathsf{I}$*, holds since* $\lambda xy.x\Omega \sqsubseteq_{\mathcal{B}} \mathsf{1} \sqsubseteq_{\mathcal{B}\eta^\infty} \mathsf{I}$ *and* $\sqsubseteq_{\mathcal{B}} \subseteq \sqsubseteq_{\mathcal{B}\eta^\infty}$*. Conclude by transitivity.*
(iv) To understand the role of the infinitary η-expansion, consider the λ-term $\mathsf{J} = \mathsf{Y}(\lambda jx.x(jx))$*. It is easy to check that* $\mathsf{J} \sqsubseteq_{\mathcal{B}\eta^\infty} \mathsf{I}$ *by looking at the in-line depiction of its Böhm tree:*

$$\mathsf{BT}(\mathsf{J}) = \lambda xy_0.x(\lambda y_1.y_0(\lambda y_2.y_1(\lambda y_3.y_2(\cdots))))$$

The relations $\sqsubseteq_{\mathcal{B}\eta^\infty}$ and $=_{\mathcal{B}\eta^\infty}$ have characterizations as contextual preorders/equivalences.

Theorem 4.7 (Hyland 1975/Wadsworth 1976). *For all λ-terms t, u, we have $t \sqsubseteq^{\mathrm{ctx}} u$ if and only if $t \sqsubseteq_{\mathcal{B}\eta^\infty} u$. Therefore $t \equiv^{\mathrm{ctx}} u$ if and only if $t =_{\mathcal{B}\eta^\infty} u$.*

5 Completeness, or Separating Böhm Different Terms

In this section, we prove that the interaction preorder $\sqsubseteq^{\mathrm{int}}$ is included in the Böhm preorder $\sqsubseteq_{\mathcal{B}}$.

Proof Technique. The standard way of proving that a contextual preorder $\sqsubseteq$ is included in a tree similarity $\precsim$ is to proceed by proving the contrapositive: one supposes $\mathsf{BT}(t) \not\precsim \mathsf{BT}(u)$, and constructs a context C that:

(1) *Extraction of the difference*: brings up this difference from possibly deep down the tree structure of $\mathsf{BT}(t)$ and $\mathsf{BT}(u)$, that is, C is such that $C\langle t\rangle$ and $C\langle u\rangle$ head reduce to two terms t' and u' for which the difference $\mathsf{BT}(t') \not\precsim \mathsf{BT}(u')$ is on the root node, and
(2) *Root separation*: exploits the root difference to make t' head terminating and u' head divergent, thus obtaining that $t \not\sqsubseteq u$.

In the literature, this extraction process is known as *Böhm out* technique [Böhm 1968]; see also [Barendregt and Manzonetto 2022; Böhm 1968; Boudol and Laneve 1996; Dezani-Ciancaglini et al. 1998; Intrigila et al. 2019]. It is a concise and yet sophisticated technique. The culprit is that the extracting context needs to first *reorganize* the applicative structure of t and u (via *tupler combinators*, see below), to then apply the suitable selectors for extracting the discriminating sub-terms.

What raises difficulties is when one has two different terms, say, $t \coloneqq xt_1t_2$ and $u \coloneqq xt_1t_3$ with the difference deep down the structure of t_2 and t_3. Intuitively, to 'extract' t_2 and t_3 one would simply substitute for x a term that selects the second argument, namely $\lambda yz.z$. Unfortunately, this approach is too simple to work, because then extracting the difference deep down t_2 and t_3 might require to substitute a *different* selecting term (say, of the first argument) for another occurrence of x in t_2 and t_3, but clearly all the occurrences of x must receive the *same* selecting term. An example of how the Böhm out technique solves this *colliding selectors issue* is discussed below.

Definitions for the Böhm Out Technique. On closed λ-terms, the technique amounts to applying the *tupler* T_n and the *i-th selector* S_i^n defined as follows:

$$\begin{array}{rrcll} n\text{-tuples} & \langle t_1,\dots,t_n\rangle & \coloneqq & \lambda x.xt_1\cdots t_n, & \text{with } x \text{ fresh;} \\ \text{Tuplers} & \mathsf{T}_n & \coloneqq & \lambda x_1\dots x_n.\langle x_1,\dots,x_n\rangle; & \\ \text{Selectors} & \mathsf{S}_i^n & \coloneqq & \lambda x_1\dots x_n.x_i, & \text{with } 1 \le i \le n. \end{array}$$

So, the tupler T_n takes n arguments $t_1,\dots,t_n$ and returns the tuple $\langle t_1,\dots,t_n\rangle$, while the selector S_i^n takes n arguments $t_1,\dots,t_n$ and returns the i-th argument t_i. Note that $\mathsf{S}_1^1 = \mathsf{I}$. Then, $\mathsf{T}_nt_1\cdots t_nu \to_{\mathsf{h}}^* ut_1\cdots t_n$ and $\mathsf{S}_i^nt_1\cdots t_n \to_{\mathsf{h}}^* t_i$, whence we have the following combined *extraction property*:

$$\mathsf{T}_nt_1\cdots t_n\mathsf{S}_i^n \quad \to_{\mathsf{h}}^* \quad t_i. \tag{1}$$

In the following, we shall need to locate nodes/hnfs occurring at a certain path in a Böhm tree.

Definition 5.1.

- Path*: a path is a (possibly empty) finite list of natural numbers $\alpha = \langle a_1,\dots,a_n\rangle$, where $a_i \ge 1$, for each $1 \le i \le n$.*
- Concatenation*: given $i \in \mathbb{N}$ and a path α as above, their concatenation is $i \cdot \alpha \coloneqq \langle i, a_1,\dots,a_n\rangle$.*
- Node occurring at a path*: let t be a λ-term such that $t \to_{\mathsf{h}}^* \lambda\vec{x}.yt_1\cdots t_k$. If $\alpha = \langle\rangle$ then $t{\restriction_\alpha} = \lambda\vec{x}.yt_1\cdots t_k$, if $\alpha = i\cdot\alpha'$ with $i \le k$ then $t_\alpha = (t_i)_{\alpha'}$. Otherwise, $t{\restriction_\alpha}$ is undefined.*

Example 5.2. *We show how the Böhm out technique is able to construct a context separating the λ-term $t \coloneqq \lambda xy.x(x\Omega y)\Omega$ from the λ-term $u \coloneqq \lambda xy.x(xy\Omega)\Omega$, which provide an example of the colliding selectors issue. We represent below their Böhm trees:*

$$\mathrm{BT}(t) = \begin{array}{ccc} & \lambda xy.x & \\ & \diagup\quad\diagdown & \\ x & & \bot \\ \diagup\quad\diagdown & & \\ \bot & & y \end{array} \qquad \mathrm{BT}(u) = \begin{array}{ccc} & \lambda xy.x & \\ & \diagup\quad\diagdown & \\ x & & \bot \\ \diagup\quad\diagdown & & \\ y & & \bot \end{array}$$

Showing that $t \not\equiv^{\mathrm{ctx}} u$ requires a context C making t converge and u diverge. The path to extract along is $\alpha' = \langle 1,2\rangle$, which showcases the colliding selectors issue: the first occurrence of x needs to select the first argument, while the second occurrence needs the second argument. Böhm's trick consists in using tuplers, as we now attempt to explain. The Böhm out context is $C \coloneqq \langle\cdot\rangle \mathsf{T}_2 \mathtt{I} \mathsf{S}_1^2 \mathsf{S}_2^2$. The idea is that, by substituting a tupler T_2 on x and having as further arguments the right selectors for each occurrence, one exploits the extraction property (1) above to select the right sub-term in each case. The identity $\mathtt{I}$ is added like a padding when more arguments are needed. Concretely, we have:

$$\begin{array}{rcl} C\langle t\rangle & = & (\lambda xy.x(x\Omega y)\Omega)\mathsf{T}_2\mathtt{I}\mathsf{S}_1^2\mathsf{S}_2^2 \\ & \to_{\mathsf{h}}^2 & \mathsf{T}_2(\mathsf{T}_2\Omega\mathtt{I})\Omega\mathsf{S}_1^2\mathsf{S}_2^2 \\ by\,(1) & \to_{\mathsf{h}}^* & \mathsf{T}_2\Omega\mathtt{I}\mathsf{S}_2^2 \\ by\,(1) & \to_{\mathsf{h}}^* & \mathtt{I} \end{array} \qquad \begin{array}{rcl} C\langle u\rangle & = & (\lambda xy.x(xy\Omega)\Omega)\mathsf{T}_2\mathtt{I}\mathsf{S}_1^2\mathsf{S}_2^2 \\ & \to_{\mathsf{h}}^2 & \mathsf{T}_2(\mathsf{T}_2\mathtt{I}\Omega)\Omega\mathsf{S}_1^2\mathsf{S}_2^2 \\ by\,(1) & \to_{\mathsf{h}}^* & \mathsf{T}_2\mathtt{I}\Omega\mathsf{S}_2^2 \\ by\,(1) & \to_{\mathsf{h}}^* & \Omega \end{array}$$

There are more technicalities of the Böhm out technique, unfortunately. Firstly, since the extraction process works through substitutions of tuplers and selectors, one in general extracts a *substitution instance of a sub-term*, and not the sub-term itself. Secondly, in our example both occurrences of x have two arguments, but in general different occurrences might have different numbers of arguments. Then, one needs to apply a tupler T_n with n "large enough", and this over-approximation may destroy some η-differences between the two trees. If one observes only termination, then η-equivalent terms are *not* separable.

In our setting, we are able to discriminate η-convertible λ-terms because we observe termination *and* count the number of interaction steps, which are changed by η, as showed by Ex. 3.6. The idea is to black dress the differing terms t and u and to white dress the separating context C, so that the black η-differences in $\overline{t}^{\bullet}$ and $\overline{u}^{\bullet}$ that are erased by $\overline{C}^{\circ}$ are turned into interaction steps.

Interaction Böhm Out. The following lemma shows how to separate those λ-terms which have Böhm trees differing *only* by some (possibly infinitary) η-expansions, that is, the case when $t \sqsubseteq_{\mathcal{B}\eta^\infty} u$ and $t \not\sqsubseteq_{\mathcal{B}} u$, because the case $t \not\sqsubseteq_{\mathcal{B}\eta^\infty} u$ is handled by Thm. 4.7, that is, via standard Böhm out. Because of its technical nature, it is labeled as a *lemma* and yet it is one of the main technical contributions of the paper. The inclusion $\sqsubseteq^{\mathrm{int}} \subseteq \sqsubseteq_{\mathcal{B}}$ then follows easily.

Lemma 5.3 (Interaction Böhm-out). *Let $t, u \in \Lambda$ such that $t \sqsubseteq_{\mathcal{B}\eta^\infty} u$ and $t \not\sqsubseteq_{\mathcal{B}} u$. Then, there exists a context $C \in \mathcal{C}$ such that $\overline{C}^{\circ}\langle \overline{t}^{\bullet}\rangle \Downarrow_{\mathsf{h}_{\circ\bullet}}^{\bullet i}$ and $\overline{C}^{\circ}\langle \overline{u}^{\bullet}\rangle \Downarrow_{\mathsf{h}_{\circ\bullet}}^{\bullet i'}$ with $i' \neq i$.*

Terminology and Notations for the Proof. As customary in mathematical analysis, we say that a relation $\mathsf{P}(-)$ holds for all $K \in \mathbb{N}$ *large enough* whenever there exists a $K' \in \mathbb{N}$ such that $\mathsf{P}(K)$ holds for all $K \geq K'$. We also use the notation $tu^{\sim n}$ for $(\cdots((tu)u)\cdots)u$ (n times). Finally, two head normal forms h, h' are *spine equivalent*, written $h =_{\mathrm{sp}} h'$, if there are $n, k \geq 0$ such that:

$$h = \lambda x_1 \ldots x_n.y\, t_1 \cdots t_k \quad \text{and} \quad h' = \lambda x_1 \ldots x_n.y\, u_1 \cdots u_k. \tag{2}$$

Proof. We prove a stronger statement, *i.e.* that there exist closed terms $\vec{s} \in \Lambda$ such that, for all $\vec{y}$ containing $\mathrm{fv}(t) \cup \mathrm{fv}(u)$ and for all $K \in \mathbb{N}$ large enough, the following holds:

$$\overline{t}^{\bullet}\{\vec{y} := \overline{\mathsf{T}_K}^{\circ}\} \circ \overline{\vec{s}}^{\circ} \Downarrow_{\mathsf{h}_{\circ\bullet}}^{\bullet i} \qquad \text{and} \qquad \overline{u}^{\bullet}\{\vec{y} := \overline{\mathsf{T}_K}^{\circ}\} \circ \overline{\vec{s}}^{\circ} \Downarrow_{\mathsf{h}_{\circ\bullet}}^{\bullet i'} \quad \text{with } i' \neq i.$$

Given variables $\vec{x}$ and a λ-term t we write $\sigma_{\vec{x}}$ for $\{\vec{x} := \overline{\mathsf{T}_K}^{\circ}\}$, and $t^{\sigma_{\vec{x}}}$ for $t\{\vec{x} := \overline{\mathsf{T}_K}^{\circ}\}$.

Note that $t \not\sqsubseteq_{\mathcal{B}} u$ is only possible if $t \Downarrow_{\mathsf{h}}$. Moreover, $t \Downarrow_{\mathsf{h}} h$ and $t \sqsubseteq_{\mathcal{B}\eta^\infty} u$ entail $u \Downarrow_{\mathsf{h}} h'$, for some h'. We proceed by induction on the length of a minimal path $\delta \in \mathbb{N}^*$ such that $t\restriction_\delta \neq_{\mathrm{sp}} u\restriction_\delta$.

Base case $\delta = \langle\rangle$, *i.e.* $h \neq_{\mathrm{sp}} h'$. Then $t \sqsubseteq_{\mathcal{B}\eta^\infty} u$ is only possible if the amount of spine abstractions and applications in h, h' can be matched via η-expansion, say:

$$t \to_{\mathsf{h}}^* h = \lambda x_1 \dots x_n . y\, t_1 \cdots t_k \qquad \text{and} \qquad u \to_{\mathsf{h}}^* h' = \lambda x_1 \dots x_n z_1 \dots z_m . y\, u_1 \cdots u_{k+m}$$

for $n, k \geq 0$ and $m > 0$. (The symmetrical case where h has more abstractions/applications than h' is omitted because analogous.) There are two subcases to consider, depending on whether y is free.

(1) *y is free, i.e. $y \in \vec{y}$.* Take any $K \geq k + m$, and empty $\vec{s}$. For t, we have:

$\bullet$-Steps	Terms and τ-steps	
	$\overline{t}^{\bullet}\{\vec{y} := \overline{\mathsf{T}_K}^{\circ}\} \to_{\mathsf{h}_\tau}^* \overline{h}^{\bullet}\{\vec{y} := \overline{\mathsf{T}_K}^{\circ}\}$,	by Pr. 3.13(1) & L.3.5
$=$	$\lambda_{\bullet\dots\bullet} x_1 \dots x_n . \overline{\mathsf{T}_K}^{\circ} \bullet \overline{t_1}^{\bullet\sigma_{\vec{y}}} \bullet \cdots \bullet \overline{t_k}^{\bullet\sigma_{\vec{y}}}$	
$\to_{\mathsf{h}_{\circ}}^k$	$\lambda_{\bullet\dots\bullet} x_1 \dots x_n . \lambda_{\circ\dots\circ} w_{k+1} \dots w_K . \langle \overline{t_1}^{\bullet\sigma_{\vec{y}}}, \dots, \overline{t_k}^{\bullet\sigma_{\vec{y}}}, w_{k+1}, \dots, w_K \rangle_{\circ}$	

where $\langle -, \dots, - \rangle_{\circ}$ denotes the tuple with white applications $\lambda z.z - \circ \cdots \circ -$. For u, we have:

$\bullet$-Steps	Terms and τ-steps	
	$\overline{u}^{\bullet}\{\vec{y} := \overline{\mathsf{T}_K}^{\circ}\} \to_{\mathsf{h}_\tau}^* \overline{h'}^{\bullet}\{\vec{y} := \overline{\mathsf{T}_K}^{\circ}\}$,	by Pr. 3.13(1) & L.3.5,
$=$	$\overline{\mathsf{T}_K}^{\circ} \bullet \overline{u_1}^{\bullet\sigma_{\vec{y}}} \bullet \cdots \bullet \overline{u_{k+m}}^{\bullet\sigma_{\vec{y}}}$	
$\to_{\mathsf{h}_{\circ}}^{k+m}$	$\lambda_{\bullet\dots\bullet} x_1 \dots x_n . \lambda_{\circ\dots\circ} w_{k+1} \dots w_K . \langle \overline{u_1}^{\bullet\sigma_{\vec{y}}}, \dots, \overline{u_{k+m}}^{\bullet\sigma_{\vec{y}}}, w_{k+m+1}, \dots, w_K \rangle_{\circ}$	

Summing up, $\overline{t}^{\bullet\sigma_{\vec{y}}} \Downarrow_{\mathsf{h}_{\circ\bullet}}^{\bullet k}$ and $\overline{u}^{\bullet\sigma_{\vec{y}}} \Downarrow_{\mathsf{h}_{\circ\bullet}}^{\bullet k+m}$. The statement holds because $m > 0$.

(2) *y is bound, i.e. $y = x_j \in \vec{x}$.* Take any $K \geq k + m$, and let the arguments $\vec{s}$ be n copies of T_K (noted $\mathsf{T}_K^{\sim n}$ for short). On the one hand:

$\bullet$-Steps	Terms and τ-steps	
	$\overline{t}^{\bullet}\{\vec{y} := \overline{\mathsf{T}_K}^{\circ}\} \circ \overline{\mathsf{T}_K}^{\circ\sim n} \to_{\mathsf{h}_\tau}^* \overline{h}^{\bullet}\{\vec{y} := \overline{\mathsf{T}_K}^{\circ}\} \circ \overline{\mathsf{T}_K}^{\circ\sim n}$,	by Pr. 3.13(1) & L.3.5,
$=$	$\left(\lambda_{\bullet\dots\bullet} x_1 \dots x_n . x_j \bullet \overline{t_1}^{\bullet\sigma_{\vec{y}}} \bullet \cdots \bullet \overline{t_k}^{\bullet\sigma_{\vec{y}}}\right) \circ \overline{\mathsf{T}_K}^{\circ\sim n}$	
$\to_{\mathsf{h}_{\bullet}}^n$	$\overline{\mathsf{T}_K}^{\circ} \bullet \overline{t_1}^{\bullet\sigma_{\vec{x}\vec{y}}} \bullet \cdots \bullet \overline{t_k}^{\bullet\sigma_{\vec{x}\vec{y}}}$	
$\to_{\mathsf{h}_{\circ}}^k$	$\lambda_{\circ\dots\circ} w_{k+1} \dots w_K . \langle \overline{t_1}^{\bullet\sigma_{\vec{x}\vec{y}}}, \dots, \overline{t_k}^{\bullet\sigma_{\vec{x}\vec{y}}}, w_{k+1}, \dots, w_K \rangle_{\circ}$	

On the other hand:

$\bullet$-Steps	Terms and τ-steps	
	$\overline{u}^{\bullet}\{\vec{y} := \overline{\mathsf{T}_K}^{\circ}\} \circ \overline{\mathsf{T}_K}^{\circ\sim n} \to_{\mathsf{h}_\tau}^* \overline{h'}^{\bullet}\{\vec{y} := \overline{\mathsf{T}_K}^{\circ}\} \circ \overline{\mathsf{T}_K}^{\circ\sim n}$,	by Pr. 3.13(1) & L.3.5,
$=$	$\left(\lambda_{\bullet\dots\bullet} x_1 \dots x_n \vec{z} . x_j \bullet \overline{u_1}^{\bullet\sigma_{\vec{y}}} \bullet \cdots \bullet \overline{u_{k+m}}^{\bullet\sigma_{\vec{y}}}\right) \circ \overline{\mathsf{T}_K}^{\circ\sim n}$	
$\to_{\mathsf{h}_{\circ\bullet}}^n$	$\overline{\mathsf{T}_K}^{\circ} \bullet \overline{u_1}^{\bullet\sigma_{\vec{x}\vec{y}}} \bullet \cdots \bullet \overline{u_{k+m}}^{\bullet\sigma_{\vec{x}\vec{y}}}$	
$\to_{\mathsf{h}_{\circ\bullet}}^{k+m}$	$\lambda_{\circ\dots\circ} w_{k+m+1} \dots w_K . \langle \overline{u_1}^{\bullet\sigma_{\vec{x}\vec{y}}}, \dots, \overline{u_{k+m}}^{\bullet\sigma_{\vec{x}\vec{y}}}, w_{k+m+1}, \dots, w_K \rangle_{\circ}$.	

Summing up, $\overline{t}^{\bullet\sigma_{\vec{y}}} \Downarrow_{\mathsf{h}_{\circ\bullet}}^{\bullet n+k}$ and $\overline{u}^{\bullet\sigma_{\vec{y}}} \Downarrow_{\mathsf{h}_{\circ\bullet}}^{\bullet n+k+m}$. The statement holds because $m > 0$.

Inductive case $\delta = j \cdot \gamma$. In this case, we must have:

$$t \to_{\mathsf{h}}^* h = \lambda x_1 \dots x_n . y\, t_1 \cdots t_k \qquad \text{and} \qquad u \to_{\mathsf{h}}^* h' = \lambda x_1 \dots x_n . y\, u_1 \cdots u_k$$

with $t_j \not\sqsubseteq_{\mathcal{B}} u_j$ and $(t_l \sqsubseteq_{\mathcal{B}\eta^\infty} u_l)_{l \leq k}$. By *i.h.*, there exists K' and $\vec{s'}$ such that for all $K \geq K'$:

$$\overline{t_j}^{\bullet\sigma_{\vec{x}\vec{y}}} \circ \overline{\vec{s'}}^{\circ} \Downarrow^{\mathfrak{o}i}_{\mathsf{h}_{\circ\bullet}} \quad \text{and} \quad \overline{u_j}^{\bullet\sigma_{\vec{x}\vec{y}}} \circ \overline{\vec{s'}}^{\circ} \Downarrow^{\mathfrak{o}i'}_{\mathsf{h}_{\circ\bullet}} \text{ with } i' \neq i.$$

We consider any $K \geq \max\{K', k\}$. We assume wlog. that y is free, the other case being analogous.

Steps	Terms
	$\overline{t}^{\bullet}\{\vec{y} := \overline{\mathsf{T}_K}^{\circ}\} \circ \overline{\mathsf{T}_K}^{\circ\sim n+K-k} \circ \overline{\mathsf{S}_j^K}^{\circ} \circ \overline{\vec{s'}}^{\circ}$, by Pr. 3.13(1) & L.3.5,
$\to^*_{\mathsf{h}_\tau}$	$\overline{h}^{\bullet}\{\vec{y} := \overline{\mathsf{T}_K}^{\circ}\} \circ \overline{\mathsf{T}_K}^{\circ\sim n+K-k} \circ \overline{\mathsf{S}_j^K}^{\circ} \circ \overline{\vec{s'}}^{\circ}$
$=$	$\left(\lambda_{\bullet\cdots\bullet} x_1 \dots x_n.\, \overline{\mathsf{T}_K}^{\circ} \bullet \overline{t_1}^{\bullet\sigma_{\vec{y}}} \bullet \cdots \bullet \overline{t_k}^{\bullet\sigma_{\vec{y}}}\right) \circ \overline{\mathsf{T}_K}^{\circ\sim n+K-k} \circ \overline{\mathsf{S}_j^K}^{\circ} \circ \overline{\vec{s'}}^{\circ}$
$\to^n_{\mathsf{h}_{\mathfrak{o}}}$	$\overline{\mathsf{T}_K}^{\circ} \bullet \overline{t_1}^{\bullet\sigma_{\vec{x}\vec{y}}} \bullet \cdots \bullet \overline{t_k}^{\bullet\sigma_{\vec{x}\vec{y}}} \circ \overline{\mathsf{T}_K}^{\circ\sim K-k} \circ \overline{\mathsf{S}_j^K}^{\circ} \circ \overline{\vec{s'}}^{\circ}$
$\to^k_{\mathsf{h}_{\mathfrak{o}}} \to^*_{\mathsf{h}_\tau}$	$\overline{t_j}^{\bullet\sigma_{\vec{x}\vec{y}}} \circ \overline{\vec{s'}}^{\circ}$ by (1).

An identical sequence of steps extracts $\overline{u_j}^{\bullet}$ from the other term, that is, we have:

$$\overline{u}^{\bullet}\{\vec{y} := \overline{\mathsf{T}_K}^{\circ}\} \circ \overline{\mathsf{T}_K}^{\circ\sim n+K-k} \circ \overline{\mathsf{S}_j^K}^{\circ} \circ \overline{\vec{s'}}^{\circ} \quad \to^*_{\mathsf{h}_\tau} \to^{n+k}_{\mathsf{h}_{\mathfrak{o}}} \to^*_{\mathsf{h}_\tau} \quad \overline{u_j}^{\bullet\sigma_{\vec{x}\vec{y}}} \circ \overline{\vec{s'}}^{\circ}$$

Note the same number of $\mathfrak{o}$-steps. By defining $\overline{\vec{s}}^{\circ}$ as the arguments $\overline{\mathsf{T}_K}^{\circ\sim n+K-k}, \overline{\mathsf{S}_j^K}^{\circ}, \overline{\vec{s'}}^{\circ}$, and by composing with what is obtained by the *i.h.*, we obtain:

$$\overline{t}^{\bullet}\{\vec{y} := \overline{\mathsf{T}_K}^{\circ}\} \circ \overline{\vec{s}}^{\circ} \Downarrow^{\mathfrak{o}n+k+i}_{\mathsf{h}_{\circ\bullet}} \quad \text{and} \quad \overline{u}^{\bullet}\{\vec{y} := \overline{\mathsf{T}_K}^{\circ}\} \circ \overline{\vec{s}}^{\circ} \Downarrow^{\mathfrak{o}n+k+i'}_{\mathsf{h}_{\circ\bullet}},$$

which is an instance of the statement because $i \neq i'$ by *i.h.* □

Theorem 5.4 (Completeness). *Let $t, u \in \Lambda$. If $t \sqsubseteq^{\text{int}} u$ then $t \sqsubseteq_{\mathcal{B}} u$.*

Proof. Assume $t \not\sqsubseteq_{\mathcal{B}} u$, towards a contradiction. There are two cases:

(i) If $t \not\sqsubseteq_{\mathcal{B}\eta^\infty} u$, then by Theorem 4.7 there exists a context C such that $C\langle t\rangle \Downarrow_{\mathsf{h}}$, while $C\langle u\rangle \not\Downarrow_{\mathsf{h}}$. By Prop. 3.14, it follows that $\overline{C}^{\circ}\langle \overline{t}^{\bullet}\rangle \Downarrow_{\mathsf{h}_{\circ\bullet}}$, while $\overline{C}^{\circ}\langle \overline{u}^{\bullet}\rangle \not\Downarrow_{\mathsf{h}_{\circ\bullet}}$. This shows $t \not\sqsubseteq^{\text{int}} u$.

(ii) If $t \sqsubseteq_{\mathcal{B}\eta^\infty} u$ then $t \not\sqsubseteq^{\text{int}} u$ follows directly from interaction Böhm out (Lemma 5.3). □

6 Multi Types and Relational Semantics

We now start preparing the ground for the proof of the inclusion $\sqsubseteq_{\mathcal{B}} \subseteq \sqsubseteq^{\text{int}}$. The main tool shall be a system of checkers multi types. The needed background on multi types, a.k.a. *non-idempotent intersection types*, is recalled here, the checkers variant shall be introduced in the next section.

Here, we present Engeler's relational model [Hyland et al. 2004] in terms of de Carvalho's system of multi types [2007; 2018], together with some classic results. In particular, we recall the multi type characterization of head normalizability, and the bounds of the length of head evaluations that can be extracted from the type derivations.

Definition 6.1. *Types and typing rules of the multi types system are given in Fig. 3.*

Multi Types. There are two categories of types: *linear types* L, which include a single[4] atomic type A and arrow types $\mathsf{M} \to \mathsf{L}$; *multi types* M, which are possibly empty multisets of linear types. Multi types are generally represented as unordered lists $[\mathsf{L}_1, \dots, \mathsf{L}_n]$ of linear types $\mathsf{L}_1, \dots, \mathsf{L}_n$, possibly with repetitions. The *empty* multi type $[\,]$, obtained by taking $n = 0$, is also denoted by $\mathbf{0}$.

A multi type $[\mathsf{L}_1, \dots, \mathsf{L}_n]$ should be intended as a conjunction $\mathsf{L}_1 \wedge \cdots \wedge \mathsf{L}_n$, for a commutative, associative, non-idempotent conjunction $\wedge$ (morally a tensor $\otimes$), having $\mathbf{0}$ as a neutral element. The intuition is that a linear type corresponds to a single use of a term t, which is typed with a multiset

[4] One may ask for more atomic types, but this choice does not really affect the results presented in the paper.

Types

$$\begin{array}{rrcl} \textsc{Linear} & \mathsf{L}, \mathsf{L}' & ::= & \mathsf{A} \mid \mathsf{M} \to \mathsf{L} \\ \textsc{Multi} & \mathsf{M}, \mathsf{N} & ::= & [\mathsf{L}_1, \dots, \mathsf{L}_n] \quad n \geq 0 \\ \textsc{Generic} & \mathsf{T}, \mathsf{T}' & ::= & \mathsf{L} \mid \mathsf{M} \\ \textsc{Zero} & \mathbf{0} & ::= & [\,] \end{array}$$

Typing rules

$$\frac{}{x : [\mathsf{L}] \vdash x : \mathsf{L}}\ \mathsf{ax} \qquad \frac{(\Gamma_i \vdash t : \mathsf{L}_i)_{i \in I} \quad I \text{ finite}}{\uplus_{i \in I} \Gamma_i \vdash t : [\mathsf{L}_i]_{i \in I}}\ \mathsf{many}$$

$$\frac{\Gamma, x : \mathsf{M} \vdash t : \mathsf{L}}{\Gamma \vdash \lambda x.t : \mathsf{M} \to \mathsf{L}}\ \lambda \qquad \frac{\Gamma \vdash t : \mathsf{M} \to \mathsf{L} \quad \Delta \vdash u : \mathsf{M}}{\Gamma \uplus \Delta \vdash tu : \mathsf{L}}\ @$$

Fig. 3. De Carvalho's multi type system.

M of cardinality n if it is going to be used n times. In particular, if $n > 0$ and t is part of a larger term u, then a copy of t shall end up in evaluation (*i.e.* head) position during the evaluation of u.

Typing Rules. Judgments have shape $\Gamma \vdash t : \mathsf{L}$ or $\Gamma \vdash t : \mathsf{M}$, where t is a λ-term, M is a multi type, L is a linear type, and Γ is a *type environment*, *i.e.*, a total function from variables to multi types such that $\mathrm{dom}(\Gamma) := \{x \mid \Gamma(x) \neq \mathbf{0}\}$ is finite. We say that Γ is *empty* if $\mathrm{dom}(\Gamma) = \emptyset$. We write $x_1 : \mathsf{M}_1, \dots, x_n : \mathsf{M}_n$ for the environment Γ such that $\Gamma(y) = \mathsf{M}_i$, if $y = x_i \in \vec{x}$, $\Gamma(y) = \mathbf{0}$, otherwise.

Note that the application rule @ requires the argument to be typed with a multi type M, which is necessarily introduced by rule many, having as hypotheses a multiset of derivations, indexed by a possibly empty set I. When I is empty, the rule @ has no premises and can type every term with $\mathbf{0}$. For instance, $\vdash \Omega : \mathbf{0}$ is derivable, but no linear type can be assigned to Ω. Intuitively, $\mathbf{0}$ is the type of erasable terms, and every λ-term is erasable in the (call-by-name) λ-calculus.

Technicalities about Types. The *multiset union* is denoted by $\uplus$ and is extended to type environments pointwisely, *i.e.* $(\Gamma \uplus \Delta)(x) := \Gamma(x) \uplus \Delta(x)$, for all $x \in \textsc{Var}$. This notion is extended further to a finite family of type environments as expected. In particular, if $J = \emptyset$ we let $\uplus_{i \in J} \Gamma_i$ be the empty environment. Given two type environments Γ and Δ having disjoint domain $\mathrm{dom}(\Gamma) \cap \mathrm{dom}(\Delta) = \emptyset$, we simply write Γ, Δ for $\Gamma \uplus \Delta$. Note that $\Gamma, x : \mathbf{0} = \Gamma$, where we implicitly assume $x \notin \mathrm{dom}(\Gamma)$. We write $\pi \rhd \Gamma \vdash t : \mathsf{T}$ whenever π is a (*type*) *derivation* (*i.e.* a finite tree constructed bottom up by applying the rules in Fig. 3) with as conclusion the judgment $\Gamma \vdash t : \mathsf{T}$. We write $\pi \rhd t$ if $\pi \rhd \Gamma \vdash t : \mathsf{T}$, for some type environment Γ and some type T.

Type Preorder and Relational Semantics. The multi type system induces a notion of semantic interpretation into what is known as *relational model* of (the call-by-name) λ-calculus. The interpretations of λ-terms are naturally ordered by set-theoretical inclusion, and this induces a preorder on λ-terms, namely the inequational theory of the model.

Definition 6.2 (Relational interpretation and type preorder).

(i) The relational interpretation $[\![t]\!]$ of a λ-term t *is defined as follows:*

$$[\![t]\!] \quad := \quad \{(\Gamma, \mathsf{L}) \mid \exists\, \pi \rhd \Gamma \vdash t : \mathsf{L}\}.$$

(ii) The type preorder $\sqsubseteq^{\mathrm{typ}}$ *on ordinary λ-terms is defined as $t \sqsubseteq^{\mathrm{typ}} u$ if $[\![t]\!] \subseteq [\![u]\!]$, and the induced* type equivalence *is noted* $\equiv^{\mathrm{typ}}$.

We have the following fundamental properties of the multi type preorder.

Theorem 6.3 (Breuvart et al. [2018]). *Let $t, u \in \Lambda$.*

(1) Compatibility*: if $t \sqsubseteq^{\mathrm{typ}} u$ then $C\langle t \rangle \sqsubseteq^{\mathrm{typ}} C\langle u \rangle$, for every context C.*
(2) β-invariance*: if $t \to_\beta u$ then $[\![t]\!] = [\![u]\!]$.*
(3) η-reduction*: if $t \to_\eta u$ then $[\![t]\!] \subseteq [\![u]\!]$.*
(4) No η-expansion*: $\lambda y.xy \to_\eta x$, but $[\![x]\!] \not\subseteq [\![\lambda y.xy]\!]$.*

The failure of η-expansion, *i.e.* Point 4 of the theorem, is due to the fact that x can be typed with the atomic type A using an axiom, while there is no way of typing $\lambda y.xy$ with A, since it can only be typed with an arrow type.

Corollary 6.4. *The relation $\sqsubseteq^{\mathrm{typ}}$ is an inequational λ-theory. Moreover, $\sqsubseteq^{\mathrm{typ}}$ is semi-extensional but not extensional.*

The corollary captures the soundness of Engeler's relational model. It is possible to prove that $\sqsubseteq^{\mathrm{typ}} \subseteq \sqsubseteq^{\mathrm{ctx}}$, thus obtaining a semantic proof that $\sqsubseteq^{\mathrm{ctx}}$ is an inequational theory (Thm. 2.6). In Sect. 8, we shall follow this approach for proving that the interaction preorder is an inequational theory.

Adequacy with Respect to Head Reduction. For showing that typable terms are head terminating, we need a notion of size for type derivations, that shall bound the number of head steps.

Definition 6.5 (Size). *Let π be a type derivation. The* (applicative) size $|\pi|_{@}$ *of π is the number of occurrences of rules @ in π.*

Proposition 6.6 ([Barendregt and Manzonetto 2022]). *Let $t, t' \in \Lambda$ be such that $t \to_{\mathsf{h}} t'$.*

(1) Quantitative subject reduction*: if $\pi \triangleright \Gamma \vdash t : \mathsf{L}$ then there exists a derivation $\pi' \triangleright \Gamma \vdash t' : \mathsf{L}$ such that $|\pi'|_{@} = |\pi|_{@} - 1$.*
(2) Subject expansion*: if $\pi' \triangleright \Gamma \vdash t' : \mathsf{L}$ then there is a derivation $\pi \triangleright \Gamma \vdash t : \mathsf{L}$.*

Note the quantitative aspect of subject reduction (Prop. 6.6(1)), stating that the derivation size *strictly decreases* along head steps. It does not say that it decreases at *arbitrary* β-steps because the contraction of redexes occurring in sub-terms typed with rule many might not change the size. For instance, if $xt \to_{\beta} xt'$ and t is typed using an empty many rule (*i.e.* with 0 premises), which is a sub-derivation of size 0, then also t' is typed using an empty many rule, of size 0. In fact, not all typable terms are β-normalizable: $x\Omega$ is typable as follows, for any linear type L, but it has no β-nf:

$$\dfrac{\dfrac{}{x : [\mathbf{0} \to \mathsf{L}] \vdash x : \mathbf{0} \to \mathsf{L}}\,\mathsf{ax} \qquad \dfrac{}{\vdash \Omega : \mathbf{0}}\,\mathsf{many}}{x : [\mathbf{0} \to \mathsf{L}] \vdash x\Omega : \mathsf{L}}\,@ \tag{3}$$

Since the size of type derivations decreases at every head step, it provides a termination measure (only) for the head reduction of typable terms. The fact that typable terms are head terminating is also called *correctness* of the type system.

Completeness of the type system—*i.e.* every head terminating term is typable—is obtained via typability of all head normal forms, proved next, and subject expansion (Prop. 6.6(2)).

Proposition 6.7 (Typability of head normal forms. [Barendregt and Manzonetto 2022]). *Let $h \in \Lambda$ be a head normal form. Then there exists a derivation $\pi \triangleright \Gamma \vdash h : \mathsf{L}$.*

Summing up, we obtain the following characterization of head normalization.

Theorem 6.8 (Typability characterizes head normalization. [Barendregt and Manzonetto 2022]). *Let $t \in \Lambda$.*

(1) Correctness*: if $\pi \triangleright t$ then there exists a head normalizing evaluation $t \to_{\mathsf{h}}^{n} h$ with h normal and $n \leq |\pi|_{@}$.*
(2) Completeness*: if $t \to_{\mathsf{h}}^{*} h$ is a head normalizing sequence, then there exists a derivation $\pi \triangleright t$.*

Therefore $[\![t]\!] \neq \emptyset$ if and only if t is head normalizable. In particular, $\sqsubseteq^{\mathrm{typ}}$ is consistent.

$$\begin{array}{rrcl} \textsc{Linear types} & \mathsf{L},\mathsf{L}' & ::= & \mathsf{A} \mid \mathsf{M} \xrightarrow{\mathsf{pq}} \mathsf{L} \qquad \mathsf{p},\mathsf{q} \in \{\circ,\bullet\} \\ \textsc{Multi types} & \mathsf{M},\mathsf{N} & ::= & [\mathsf{L}_1,\dots,\mathsf{L}_n] \qquad n \geq 0 \end{array} \quad \Big| \quad \begin{array}{rrcl} \textsc{Generic types} & \mathsf{T},\mathsf{T}' & ::= & \mathsf{L} \mid \mathsf{M} \\ \textsc{Zero} & \mathbf{0} & ::= & [\,] \end{array}$$

$$\frac{}{x:[\mathsf{L}] \vdash^{0}_{\bullet} x:\mathsf{L}}\ \mathsf{ax} \qquad \frac{(\Gamma_i \vdash^{k_i}_{\bullet} t:\mathsf{L}_i)_{i\in I} \quad I \text{ finite}}{\uplus_{i\in I}\Gamma_i \vdash^{\sum_{i\in I} k_i}_{\bullet} t:[\mathsf{L}_i]_{i\in I}}\ \mathsf{many}$$

$$\frac{\Gamma, x:\mathsf{M} \vdash^{k}_{\bullet} t:\mathsf{L}}{\Gamma \vdash^{k}_{\bullet} \lambda_{\mathsf{p}} x.t:\mathsf{M} \xrightarrow{\mathsf{pq}} \mathsf{L}}\ \lambda \qquad \frac{\Gamma \vdash^{k_1}_{\bullet} t:\mathsf{M} \xrightarrow{\mathsf{pq}} \mathsf{L} \quad \Delta \vdash^{k_2}_{\bullet} u:\mathsf{M}}{\Gamma \uplus \Delta \vdash^{k}_{\bullet} t \cdot^{\mathsf{q}} u:\mathsf{L}}\ @$$

$$\frac{\Gamma \vdash^{k_1}_{\bullet} t:\mathsf{M} \xrightarrow{\mathsf{pp}} \mathsf{L} \quad \Delta \vdash^{k_2}_{\bullet} u:\mathsf{M}}{\Gamma \uplus \Delta \vdash^{k_1+k_2}_{\bullet} t \cdot^{\mathsf{p}} u:\mathsf{L}}\ @_\tau \qquad \frac{\Gamma \vdash^{k_1}_{\bullet} t:\mathsf{M} \xrightarrow{\mathsf{pp}^\perp} \mathsf{L} \quad \Delta \vdash^{k_2}_{\bullet} u:\mathsf{M}}{\Gamma \uplus \Delta \vdash^{k_1+k_2+1}_{\bullet} t \cdot^{\mathsf{p}^\perp} u:\mathsf{L}}\ @_\bullet$$

In rule @, $k = k_1+k_2$ if $\mathsf{p} = \mathsf{q}$, otherwise $k = k_1+k_2+1$. Rule @ compactly sums up rules $@_\tau$ and $@_\bullet$.

Fig. 4. Checkers multi type system $\vdash_{\bullet}$.

Exact Bounds? It is natural to wonder whether there are type derivations π for which correctness holds with $n = |\pi|_{@}$. The answer is no: the type derivation in (3), as well as any other type derivation for $x\Omega$, has at least one @ rule even if the λ-term $x\Omega$ is already a head normal form, whence $n = 0$.

The question has been studied and refined in the literature. Such a mismatch can be improved in two ways, both studied in-depth by Accattoli et al. [2020]. The first one traces back to de Carvalho [2007, 2018], and takes into account the number $|h|_{\mathsf{h}}$ of application constructors in the spine of the head normal form h. Then type derivations π satisfying a certain *tight predicate* verify $n+|h|_{\mathsf{h}} = |\pi|_{@}$. The second one is developed in Accattoli et al. [2020]. It introduces:

- A second set of typing rules assigning some new type constants to the constructors that occur in the head normal form, and;
- A tight predicate forcing all such constructors to be typed with these alternative rules.

In such a system, one can actually obtain $n = |\pi|_{@}$ when the tight predicate holds. The drawback is that in this case one obtains a constants-only type that cannot be composed with any other type.

For the new type system of the next section, we shall give in Sect. 8 a refined technique for exact bounds that exploits player tags. We shall measure the exact number of interaction head steps without resorting to constants-only types, which is a novelty.

7 Checkers Multi Types

In this section, we introduce a system $\vdash_{\bullet}$ of multi types for the checkers calculus, that can be seen as an annotated version of the standard one presented in Section 6. We shall prove that the new system characterizes termination of checkers head reduction $\to_{\mathsf{h}_{\circ\bullet}}$, similarly to the standard system. Despite the similarity, however, the two systems are inherently different because the new one shall *not* be invariant under η-reduction, while the standard one is (Thm. 6.3(3)).

Definition 7.1. *Types and typing rules of the checkers multi types system $\vdash_{\bullet}$ are given in Fig. 4.*

Main Ideas 1: Checkers Arrows. We start by turning the arrow type $\mathsf{M} \to \mathsf{L}$ into a checkers arrow type $\mathsf{M} \xrightarrow{\mathsf{pq}} \mathsf{L}$ carrying *two* tags $\mathsf{p},\mathsf{q} \in \{\bullet,\circ\}$, thus giving rise to four possible player combinations. The idea is that if, say, $t : \mathsf{M} \xrightarrow{\circ\bullet} \mathsf{L}$ then t can only be applied via $\bullet$-applications and if it reduces to an abstraction then it must be a $\circ$-abstraction. In the typing rule λ for abstractions, the first player p is determined by the external abstraction, while the second player q can be freely chosen. We shall refer to $\xrightarrow{\circ\bullet}$ and $\xrightarrow{\bullet\circ}$ as *interaction arrow types*, and to $\xrightarrow{\bullet\bullet}$ and $\xrightarrow{\circ\circ}$ as *silent arrow types*.

Main Ideas 2: Interaction Application Rule and Index. The second main ingredient is that there are now two application rules $@_\tau$ and $@_\bullet$, one for the application of silent arrows and one for interaction arrows. We provide also a rule @ that sums them up compactly. Moreover, typing judgement $\Gamma \vdash^k_\bullet t : \mathsf{T}$ now carry an index k which counts the number of $@_\bullet$ rules in the derivation. These rules, intuitively, type interaction steps, which can be *factual* or *potential*, as we now explain. The application of an abstraction typed with an interaction arrow type gives rise to an interaction step $\to_{\beta_\bullet}$, whence it is a factual interaction. The application of a free variable typed with an interaction arrow, for instance, does *not* give rise to an interaction step, but that *potential* interaction is recorded in the type and it might arise if the term is plugged in a context, according to its type.

When the interaction index k is irrelevant, we omit it and simply write $\Gamma \vdash_\bullet t : \mathsf{T}$.

Definition 7.2 (Checkers relational interpretation and type preorder).

(i) The checkers relational interpretation $[\![t]\!]^\bullet$ of a checkers term $t \in \Lambda_{\circ\bullet}$ *is defined by:*

$$[\![t]\!]^\bullet \quad := \quad \{((\Gamma, k, \mathsf{L}) \mid \exists\, \pi \triangleright \Gamma \vdash^k_\bullet t : \mathsf{L}\}.$$

(ii) The type preorder $\sqsubseteq^{\mathrm{typ}}_\bullet$ *on checkers* λ*-terms* $t, u \in \Lambda_{\circ\bullet}$ *is defined as* $t \sqsubseteq^{\mathrm{typ}}_\bullet u$ *if* $[\![t]\!]^\bullet \subseteq [\![u]\!]^\bullet$*, and the induced type equivalence is noted* $\equiv^{\mathrm{typ}}_\bullet$.

(iii) The black type preorder $\sqsubseteq^{\bullet\mathrm{typ}}$ *on ordinary* λ*-terms* $t, u \in \Lambda$ *is defined as* $t \sqsubseteq^{\bullet\mathrm{typ}} u$ *if* $\overline{t}^\bullet \sqsubseteq^{\mathrm{typ}}_\bullet \overline{u}^\bullet$*, and the induced type equivalence is noted* $\equiv^{\bullet\mathrm{typ}}$.

We have the following fundamental properties of the multi type preorder.

Theorem 7.3. *Let* $t, u \in \Lambda_{\circ\bullet}$.

(1) Compatibility*: if* $t \sqsubseteq^{\mathrm{typ}}_\bullet u$ *then* $C\langle t\rangle \sqsubseteq^{\mathrm{typ}}_\bullet C\langle u\rangle$*, for every context* C.
(2) Silent β-invariance*: if* $t \to_{\beta_\tau} u$ *then* $[\![t]\!]^\bullet = [\![u]\!]^\bullet$.
(3) No η-reduction*: for all players* $\mathsf{p}, \mathsf{q} \in \{\bullet, \circ\}$. $[\![\lambda_{\mathsf{p}} y.x \cdot^{\mathsf{q}} y]\!]^\bullet \not\subseteq [\![x]\!]^\bullet$.
(4) No η-expansion*: for all players* $\mathsf{p}, \mathsf{q} \in \{\bullet, \circ\}$. $[\![x]\!]^\bullet \not\subseteq [\![\lambda_{\mathsf{p}} y.x \cdot^{\mathsf{q}} y]\!]^\bullet$.

The fact that the checkers relational interpretation invalidates η-reduction, is specific to interaction arrow types. As an example of Thm. 7.3(3), consider the black η-expansion of x:

$$\dfrac{\dfrac{\dfrac{}{x : [\mathbf{0} \xrightarrow{\circ\bullet} \mathsf{L}] \vdash^0_\bullet x : \mathbf{0} \xrightarrow{\circ\bullet} \mathsf{L}}\,\mathsf{ax} \qquad \dfrac{}{\vdash^0_\bullet y : \mathbf{0}}\,\mathsf{many}}{x : [\mathbf{0} \xrightarrow{\circ\bullet} \mathsf{L}] \vdash^1_\bullet x \bullet y : \mathsf{L}}\,@_\bullet}{x : [\mathbf{0} \xrightarrow{\circ\bullet} \mathsf{L}] \vdash^1_\bullet \lambda_\bullet y.x \bullet y : \mathbf{0} \xrightarrow{\bullet\mathsf{p}} \mathsf{L}}\,\lambda$$

and note that instead $x : [\mathbf{0} \xrightarrow{\circ\bullet} \mathsf{L}] \not\vdash^k_\bullet x : \mathbf{0} \xrightarrow{\bullet\mathsf{p}} \mathsf{L}$, for all indices k and players p. An analogous typing derivation shows that $\lambda_\bullet y.x \circ y \not\sqsubseteq^{\mathrm{typ}}_\bullet x$.

Corollary 7.4. *The black type preorder* $\sqsubseteq^{\bullet\mathrm{typ}}$ *is an inequational* λ*-theory. Moreover,* $\sqsubseteq^{\bullet\mathrm{typ}}$ *is neither extensional nor semi-extensional.*

Adequacy with Respect to Checkers Head Reduction. As in the plain type system, the applicative size $|\pi|_@$ of type derivations (which is still defined as the number of rules @ in π, even if the rule itself has changed) decreases with each head step $\to_{\mathsf{h}_{\circ\bullet}}$. The difference, however, is that if the step is an interaction one—and only in that case—then also the index k decreases by exactly 1.

Proposition 7.5. *Let* $t, t' \in \Lambda_{\circ\bullet}$ *be such that* $t \to_{\mathsf{h}_{\circ\bullet}} t'$.

(1) Quantitative subject reduction*: if* $\pi \triangleright \Gamma \vdash^k_\bullet t : \mathsf{L}$ *then there is a derivation* $\pi' \triangleright \Gamma \vdash^{k'}_\bullet t' : \mathsf{L}$ *such that* $|\pi'|_@ = |\pi|_@ - 1$*. Moreover, if* $t \to_{\mathsf{h}_\bullet} t'$ *then* $k' = k - 1$ *and if* $t \to_{\mathsf{h}_\tau} t'$ *then* $k' = k$.
(2) Subject expansion*: if* $\pi' \triangleright \Gamma \vdash_\bullet t' : \mathsf{L}$ *then there is a derivation* $\pi \triangleright \Gamma \vdash_\bullet t : \mathsf{L}$.

As before, quantitative subject reduction entails the correctness of the checkers type system, and the following typability of head normal forms gives its completeness.

Proposition 7.6 (Typability of head normal forms). *Let $h \in \Lambda_{\circ\bullet}$ be a head normal form. Then there exists a derivation $\pi \triangleright \Gamma \vdash^{\bullet}_{\bullet} h : \mathsf{L}$.*

Typability of all head normal forms (Prop. 7.6) together with subject expansion (Prop. 7.5(2)) implies the *completeness* of the type system: every head terminating term is typable. Summing up, we obtain the following characterization of head normalization.

Theorem 7.7 (Typability characterizes head normalization). *Let $t \in \Lambda_{\circ\bullet}$.*

(1) Correctness*: if $\pi \triangleright \Gamma \vdash^{k}_{\bullet} t : \mathsf{L}$ then there exists a $\mathsf{h}_{\circ\bullet}$-normal form h and an evaluation sequence $e : t \to^{n}_{\mathsf{h}_{\circ\bullet}} h$ with $n \leq |\pi|_{@}$ and such that the number of interaction steps in e is $\leq k$.*

(2) Completeness*: if $t \to^{*}_{\mathsf{h}_{\circ\bullet}} h$ is a head normalizing sequence, then there exists $\pi \triangleright \Gamma \vdash^{\bullet}_{\bullet} t : \mathsf{L}$.*

Therefore, $[\![t]\!]^{\bullet} \neq \emptyset$ if and only if t is $\mathsf{h}_{\circ\bullet}$-normalizable.

8 From the Böhm Preorder to the Interaction One, via Multi Types

In this section, we use the obtained results about checkers multi types to prove the chain of inclusions $\sqsubseteq_{\mathcal{B}} \subseteq \sqsubseteq^{\bullet\mathrm{typ}} \subseteq \sqsubseteq^{\mathrm{int}}$, as to complete the proof that $\sqsubseteq_{\mathcal{B}} = \sqsubseteq^{\mathrm{int}}$. The first inclusion $\sqsubseteq_{\mathcal{B}} \subseteq \sqsubseteq^{\bullet\mathrm{typ}}$ is simple: it follows from an easy induction on the size of checkers type derivations, exploiting the properties of the checkers type system. The second inclusion $\sqsubseteq^{\bullet\mathrm{typ}} \subseteq \sqsubseteq^{\mathrm{int}}$ requires slightly more work. The key point is to show that the type preorder preserves the number of interaction steps during head normalization in $\Lambda_{\circ\bullet}$, that is, that if $t \Downarrow^{\bullet k}_{\mathsf{h}_{\circ\bullet}}$ and $t \sqsubseteq^{\mathrm{typ}}_{\bullet} u$ then $u \Downarrow^{\bullet k}_{\mathsf{h}_{\circ\bullet}}$. Such a property requires to characterize a class of checkers type derivations whose index captures exactly the number of head interaction steps to head normal form. We do it via a notion of *tight* typing.

The First Inclusion. The proof of the following proposition goes by a simple induction on the size of derivations. The use of *quantitative* subject reduction (hence of multi types, instead of idempotent intersection types) is critical in order for the induction argument to go through.

Theorem 8.1 (The Böhm preorder is included in the checkers type preorder). *Let $t, u \in \Lambda$. If $t \sqsubseteq_{\mathcal{B}} u$ then $t \sqsubseteq^{\bullet\mathrm{typ}} u$.*

Proof. Assume $t \sqsubseteq_{\mathcal{B}} u$. If $t \not\Downarrow_{\mathsf{h}}$ then by Proposition 3.13 also $\overline{t}^{\bullet}$ is not $\mathsf{h}_{\circ\bullet}$-normalizable. From Thm. 7.7, we obtain $[\![\overline{t}^{\bullet}]\!]^{\bullet} = \emptyset \subseteq [\![\overline{u}^{\bullet}]\!]^{\bullet}$ whence $t \sqsubseteq^{\bullet\mathrm{typ}} u$ (by definition).

Otherwise, we must have $t \Downarrow_{\mathsf{h}} h_t \coloneqq \lambda x_1 \dots x_m.y\, t_1 \cdots t_n$ and $u \Downarrow_{\mathsf{h}} h_u \coloneqq \lambda x_1 \dots x_m.y\, u_1 \cdots u_n$, with $(t_i \sqsubseteq_{\mathcal{B}} u_i)_{i \leq n}$. Let us take any typing $(\Gamma, k, \mathsf{L}) \in [\![\overline{t}^{\bullet}]\!]^{\bullet}$, and show that it belongs to $[\![\overline{u}^{\bullet}]\!]^{\bullet}$. We proceed by induction on the size $|\pi|_{@}$ of a derivation $\pi \triangleright \Gamma \vdash^{k}_{\bullet} \overline{t}^{\bullet} : \mathsf{L}$. By Prop. 3.13(1) we get $\overline{t}^{\bullet} \to^{*}_{\mathsf{h}_\tau} \overline{h_t}^{\bullet}$, and by quantitative subject reduction (Prop. 7.5(1)) there is a derivation $\pi_{\mathrm{hnf}} \triangleright \vdash^{k'}_{\bullet} \overline{h_t}^{\bullet} : \mathsf{L}$ such that $k = k'$ (since all head reductions involved are silent) and $|\pi_{\mathrm{hnf}}|_{@} \leq |\pi|_{@}$. Moreover,

$$\pi_{\mathrm{hnf}} \quad = \quad \dfrac{\dfrac{\dfrac{\Delta_0 \vdash^{0}_{\bullet} y : \mathsf{N}_1 \xrightarrow{\mathsf{p}_1^{\bullet}} \cdots \mathsf{N}_n \xrightarrow{\mathsf{p}_n^{\bullet}} \mathsf{L}' \quad (\pi_i \triangleright \Delta_i \vdash^{k_i}_{\bullet} \overline{t_i}^{\bullet} : \mathsf{N}_i)_{1 \leq i \leq n}}{\Gamma, (x_i : \mathsf{M}_i)_{1 \leq i \leq m} \vdash^{k}_{\bullet} \overline{y\, t_1 \cdots t_n}^{\bullet} : \mathsf{L}'}\, n\,@}{\Gamma \vdash^{k}_{\bullet} \overline{h_t}^{\bullet} : \mathsf{L} = \mathsf{M}_1 \xrightarrow{\bullet \mathsf{q}_1} \cdots \mathsf{M}_n \xrightarrow{\bullet \mathsf{q}_m} \mathsf{L}'}\, m\,\lambda$$

where $n\,@$ (resp. $m\,\lambda$) denotes n (resp. m) consecutive applications of the rule, and

- $\Gamma, (x_i : \mathsf{M}_i)_{1 \leq i \leq m} = \biguplus_{0 \leq i \leq n} \Delta_i$;
- $\sum_{1 \leq i \leq n} (k_i + \gamma_{\bullet}(\mathsf{p}_i)) = k$, where $\gamma_{\bullet}(a) \coloneqq 0$ if $a = \bullet$, and $\gamma_{\bullet}(a) \coloneqq 1$ otherwise.

Note that, for every $i \leq n$, $t_i \sqsubseteq_{\mathcal{B}} u_i$ and $|\pi_i|_{@} < |\pi_{\mathrm{hnf}}|_{@} \leq |\pi|_{@}$, so we can apply the *i.h.* to each linear type in the multi type N_i and get a derivation $\Delta_i \vdash^{k_i}_{\mathcal{O}} \overline{u_i}^{\bullet} : \mathsf{N}_i$. We use these derivations to build the appropriate derivation for h_u:

$$\dfrac{\dfrac{\Delta_0 \vdash^{0}_{\mathcal{O}} y : \mathsf{N}_1 \xrightarrow{\mathrm{p}_1^{\bullet}} \cdots \mathsf{N}_n \xrightarrow{\mathrm{p}_n^{\bullet}} \mathsf{L}' \quad (\pi_{u_i} \triangleright \Delta_i \vdash^{k_i}_{\mathcal{O}} \overline{u_i}^{\bullet} : \mathsf{N}_i)_{1 \leq i \leq n}}{\Gamma, (x_i : \mathsf{M}_i)_{1 \leq i \leq m} \vdash^{k}_{\mathcal{O}} \overline{y\, u_1 \cdots u_n}^{\bullet} : \mathsf{L}'}\, n\,@}{\Gamma \vdash^{k}_{\mathcal{O}} \overline{h_u}^{\bullet} : \mathsf{L} = \mathsf{M}_1 \xrightarrow{\bullet \mathrm{q}_1} \cdots \mathsf{M}_n \xrightarrow{\bullet \mathrm{q}_m} \mathsf{L}'}\, m\,\lambda$$

By Prop. 3.13(1) we get $\overline{u}^{\bullet} \to^{*}_{\mathsf{h}_\tau} \overline{h_u}^{\bullet}$ and by subject expansion (Prop. 7.5(2)) we obtain $\Gamma \vdash^{k'}_{\mathcal{O}} \overline{u}^{\bullet} : \mathsf{L}$. Since all head reductions involved are silent, we conclude $k = k'$ and $(\Gamma, k, \mathsf{L}) \in [\![\overline{u}^{\bullet}]\!]^{\mathcal{O}}$.

(Note that a 'hidden' base case is when $n = 0$, as it does not require the induction hypothesis.) □

The Second Inclusion, via Tight Typings. Now, we consider a *tight* predicate on type judgements that forces the index k on the type derivation to be exactly the number of head interaction steps to head normal form, as we show below. An essential aspect is the fact that the predicate concerns *types*, and not *type derivations*, so that it can be transferred from t to u when they are related by the type preorder, that is, when $t \sqsubseteq^{\mathrm{typ}}_{\mathcal{O}} u$.

Definition 8.2 (Tight typings and derivations). *Let $t \in \Lambda_{\circ\bullet}$ and $\Gamma \vdash_{\mathcal{O}} t : \mathsf{L}$ be a type judgement. The pair (Γ, L) is a* tight typing *if:*

(1) All multi types M occurring in Γ and L are empty except for one, and
(2) All arrows in (Γ, L) are silent.

For ease of language, we shall also say that a derivation $\pi \triangleright \Gamma \vdash_{\mathcal{O}} t : \mathsf{L}$ is tight if (Γ, L) is a tight typing.

The definition of tight typing can actually be slightly weakened, by asking that only the arrows in the non-empty multi type are silent, without loosing any of its properties. This weakened definition, however, is slightly more technical, which is why we avoid it.

The crucial property ensured by tightness is that tightly typed head normal forms have interaction index 0, and that a tight typing can be derived for every head normal form.

Proposition 8.3 (Tightness and head normal forms). *Let $h \in \Lambda_{\circ\bullet}$ be a $\mathsf{h}_{\circ\bullet}$-normal form.*

(1) Existence: *there exists a tight derivation $\Gamma \vdash^{k}_{\mathcal{O}} h : \mathsf{L}$;*
(2) Zero interaction: *if $\Gamma \vdash^{k}_{\mathcal{O}} h : \mathsf{L}$ is tight then $k = 0$.*

From the properties of tightness for head normal forms and the characterization of head reduction (Thm. 7.7), we obtain the following refined characterization.

Theorem 8.4 (Tight characterization). *Let $t \in \Lambda_{\circ\bullet}$.*

(1) Correctness*: if $\pi : \Gamma \vdash^{k}_{\mathcal{O}} t : \mathsf{L}$ is tight then there exists a head normal form h and an evaluation sequence $e : t \to^{*}_{\mathsf{h}_{\circ\bullet}} h$ such that the number of interaction steps in e is exactly k.*
(2) Completeness*: if $e : t \to^{*}_{\mathsf{h}_{\circ\bullet}} h$ with h head normal, and k is the number of interaction steps in e, then there exists a tight derivation $\pi : \Gamma \vdash^{k}_{\mathcal{O}} t : \mathsf{L}$.*

Proof.

(1) By correctness (Thm. 7.7(1)), there exists an evaluation sequence $e : t \to^{*}_{\mathsf{h}_{\circ\bullet}} h$ such that the number of interaction steps in e is $k' \leq k$. By quantitative subject reduction (Prop. 7.5(1)), $\Gamma \vdash^{k-k'}_{\mathcal{O}} h : \mathsf{L}$. Since (Γ, L) is tight, by the zero interaction property of tight typings (Prop. 8.3(2)) we obtain $k - k' = 0$, that is, $k' = k$.

(2) By the existence of tight derivations for head normal forms (Prop. 8.3(1)), we obtain a tight derivation $\pi \triangleright \Gamma \vdash^{0}_{\mathcal{O}} h : \mathsf{L}$. By subject expansion (Prop. 7.5), the same typing types t but with an index k', *i.e.* $\Gamma \vdash^{k'}_{\mathcal{O}} t : \mathsf{L}$. By Point 1, $t \to^{*}_{\mathsf{h}_{\circ\bullet}} h'$ for some head normal form h' doing k' interaction steps. By determinism of $\to_{\mathsf{h}_{\circ\bullet}}$, $h = h'$ and $k = k'$. □

The tight characterization is then used to show that the type preorder is sound with respect to the interaction preorder.

Corollary 8.5.

(1) Tightness of the checkers type preorder*: let* $t, u \in \Lambda_{\circ\bullet}$*. If* $t \Downarrow^{\mathcal{O}k}_{\mathsf{h}_{\circ\bullet}}$ *and* $t \sqsubseteq^{\mathrm{typ}}_{\mathcal{O}} u$ *then* $u \Downarrow^{\mathcal{O}k}_{\mathsf{h}_{\circ\bullet}}$.

(2) Soundness of the checkers type preorder*: let* $t, u \in \Lambda_{\circ\bullet}$*. If* $t \sqsubseteq^{\mathrm{typ}}_{\mathcal{O}} u$ *then* $t \sqsubseteq^{\mathrm{ctx}}_{\mathcal{O}} u$.

(3) On ordinary λ-terms*: let* $t, u \in \Lambda$*. If* $t \sqsubseteq^{\bullet\mathrm{typ}} u$ *then* $t \sqsubseteq^{\mathrm{int}} u$.

Proof. (3) follows immediately from (2).

(1) Let t and u such that $t \sqsubseteq^{\mathrm{typ}}_{\mathcal{O}} u$ and $t \Downarrow^{\mathcal{O}k}_{\mathsf{h}_{\circ\bullet}}$. By tight completeness (Thm. 8.4(2)), there exists a tight typing Γ, L such that $\Gamma \vdash^{k}_{\mathcal{O}} t : \mathsf{L}$. By $t \sqsubseteq^{\mathrm{typ}}_{\mathcal{O}} u$, we obtain a derivation of $\Gamma \vdash^{k}_{\mathcal{O}} u : \mathsf{L}$. By tight correctness (Thm. 8.4(1)) and tightness of (Γ, L), we obtain $u \Downarrow^{\mathcal{O}k}_{\mathsf{h}_{\circ\bullet}}$.

(2) Let $t, u \in \Lambda_{\circ\bullet}$ such that $t \sqsubseteq^{\mathrm{typ}}_{\mathcal{O}} u$. By compatibility of $\sqsubseteq^{\mathrm{typ}}_{\mathcal{O}}$ (Thm. 7.3(1)), $C\langle t\rangle \sqsubseteq^{\mathrm{typ}}_{\mathcal{O}} C\langle u\rangle$ for all $C \in \mathcal{C}_{\circ\bullet}$. By tightness of $\sqsubseteq^{\mathrm{typ}}_{\mathcal{O}}$ (Point 1), if $C\langle t\rangle \Downarrow^{\mathcal{O}k}_{\mathsf{h}_{\circ\bullet}}$ then $C\langle u\rangle \Downarrow^{\mathcal{O}k}_{\mathsf{h}_{\circ\bullet}}$. Hence, $t \sqsubseteq^{\mathrm{ctx}}_{\mathcal{O}} u$. □

We can now put the all the inclusions together, thus obtaining our main theorem.

Theorem 8.6 (Tree and type characterizations of interaction equivalence). *The preorders* $\sqsubseteq_{\mathcal{B}}$, $\sqsubseteq^{\bullet\mathrm{typ}}$, *and* $\sqsubseteq^{\mathrm{int}}$ *coincide. Therefore, the* λ*-theories* $=_{\mathcal{B}}$, $\equiv^{\bullet\mathrm{typ}}$, *and* $\equiv^{\mathrm{int}}$ *coincide.*

Proof. ($\sqsubseteq_{\mathcal{B}} \subseteq \sqsubseteq^{\bullet\mathrm{typ}}$) By Thm. 8.1. ($\sqsubseteq^{\bullet\mathrm{typ}} \subseteq \sqsubseteq^{\mathrm{int}}$) By Cor. 8.5(3). ($\sqsubseteq^{\mathrm{int}} \subseteq \sqsubseteq_{\mathcal{B}}$) By Thm. 5.4. □

Back to a Delayed Proof. We can now finally prove Thm. 3.11, stating that silent conversion $=_{\beta_\tau}$ is included in the checkers interaction preorder $\sqsubseteq^{\mathrm{int}}$, which is the key point of the proof that the interaction preorder $\sqsubseteq^{\mathrm{int}}$ is an inequational λ-theory (Cor. 3.12).

Theorem 3.11 (The interaction preorder includes silent conversion). *For all checkers terms* $t, u \in \Lambda_{\circ\bullet}$, $t =_{\beta_\tau} u$ *entails* $t \sqsubseteq^{\mathrm{ctx}}_{\mathcal{O}} u$.

Proof. From Thm. 7.3(2), if $t =_{\beta_\tau} u$ then $t \sqsubseteq^{\mathrm{typ}}_{\mathcal{O}} u$. By Cor. 8.5(2), we obtain $t \sqsubseteq^{\mathrm{ctx}}_{\mathcal{O}} u$. □

Interaction Improvement and η. We provided Böhm tree characterizations of $\sqsubseteq^{\mathrm{int}}$ and $\equiv^{\mathrm{int}}$, but not of the interaction improvement $\sqsubseteq^{\mathrm{int\cdot imp}}$. Nonetheless, we almost have one. In the previous sections, we obtained the following chain of relationships:

$$\sqsubseteq_{\mathcal{B}} \quad =_{T.8.6} \quad \sqsubseteq^{\mathrm{int}} \quad \subsetneq_{L.3.16} \quad \sqsubseteq^{\mathrm{int\cdot imp}} \quad \subsetneq_{L.3.16} \quad \sqsubseteq^{\mathrm{ctx}} \quad =_{T.4.7} \quad \sqsubseteq_{\mathcal{B}\eta^{\infty}} \tag{4}$$

That is, interaction improvement $\sqsubseteq^{\mathrm{int\cdot imp}}$ possibly enlarges the interaction preorder $\sqsubseteq^{\mathrm{int}}$ and yet stays confined within the further fence of η-equivalence. Additionally, the fact that η-reduction can decrease the number of interactions (Ex. 3.6) suggests the following.

Conjecture 8.7. *Interaction improvement* $\sqsubseteq^{\mathrm{int\cdot imp}}$ *is characterized by* $\sqsubseteq_{\mathcal{B}\eta^{\infty}_{\mathrm{red}}}$*, the variant of* $\sqsubseteq_{\mathcal{B}\eta^{\infty}}$ *(Definition 4.5) up to possibly infinite* η-reduction *(rather than* η-equivalence*).*

Breuvart et al. [2018] prove that the preorder $\sqsubseteq_{\mathcal{B}\eta^{\infty}_{\mathrm{red}}}$ coincides with the preorder $\sqsubseteq^{\mathrm{typ}}$ induced by (plain) multi types (Definition 6.2), which validates η-reduction (Thm. 6.3.3). As already mentioned, the difficulty for proving the conjecture is managing η-reduction in the checkers framework, since both rewriting techniques and checkers multi types fail to handle it. Actually, the variant of Böhm

out technique of Section 5 smoothly adapts from preorder to improvement, giving $\sqsubseteq^{\mathrm{int}\cdot\mathrm{imp}} \subseteq \sqsubseteq_{\mathcal{B}\eta^{\infty}_{\mathrm{red}}}$, that is, $\sqsubseteq_{\mathcal{B}} \subseteq \sqsubseteq^{\mathrm{int}\cdot\mathrm{imp}} \subseteq \sqsubseteq_{\mathcal{B}\eta^{\infty}_{\mathrm{red}}}$. Therefore, conjecture 8.7 reduces to prove $\sqsubseteq_{\mathcal{B}\eta^{\infty}_{\mathrm{red}}} \subseteq \sqsubseteq^{\mathrm{int}\cdot\mathrm{imp}}$.

White Contexts Are Enough. In the proof of the completeness theorem (Thm. 5.4), black terms are separated using only white contexts. The next corollary guarantees that, for the interaction preorder, white contexts are as discriminating as general checkers contexts. Such a strong fact validates the intuition that the interaction preorder amounts to consider the program and the context as *different* players.

Corollary 8.8 (The interaction preorder can be restricted to white contexts). *Let $t, u \in \Lambda$ and the preorder $t \sqsubseteq^{\circ\mathrm{int}} u$ be defined as: for all ordinary contexts $C \in \mathcal{C}$, if there exists k such that $\overline{C}^{\circ}\langle \overline{t}^{\bullet}\rangle \Downarrow^{\bullet k}_{\mathrm{h}_{\circ\bullet}}$ then $\overline{C}^{\circ}\langle \overline{u}^{\bullet}\rangle \Downarrow^{\bullet k}_{\mathrm{h}_{\circ\bullet}}$. Then, $t \sqsubseteq^{\mathrm{int}} u$ if and only if $t \sqsubseteq^{\circ\mathrm{int}} u$.*

Proof. Thm. 5.4 shows that $t \sqsubseteq^{\circ\mathrm{int}} u$ implies $t \sqsubseteq_{\mathcal{B}} u$. Thm. 8.6 shows that $t \sqsubseteq_{\mathcal{B}} u$ implies $t \sqsubseteq^{\mathrm{int}} u$. Clearly $t \sqsubseteq^{\mathrm{int}} u$ implies $t \sqsubseteq^{\circ\mathrm{int}} u$, as $\overline{C}^{\circ}$ is a checkers context for any $C \in \mathcal{C}$. □

9 Related Work

Improvements. Improvements [Sands 1996a,b, 1999] were developed in the '90s by Sands and co-authors to prove that various program transformations are time or space improvements [Gustavsson and Sands 2001; Moran and Sands 1999; Sands 1996a], in the context of call-by-need evaluation, and have seen a revival in recent years [Hackett and Hutton 2014, 2015, 2018; Muroya and Hamana 2024; Riely and Prins 2000; Schmidt-Schauß et al. 2018].

Observational Equivalences and Trees. The characterization of observational equivalences in terms of equalities on trees originates in [Hyland 1975, 1976; Wadsworth 1976], and is an ongoing line of research whose state of the art is presented in [Intrigila et al. 2019]. The question appears in the TLCA list of open problems [Dezani-Ciancaglini 2001]. This paper contributes to this line of research, somewhat backwards: we introduce a new observational equivalence that matches a known equality on trees, namely non-extensional Böhm tree equality $\mathcal{B}$.

Böhm Tree Equality. For the specific case of $\mathcal{B}$, the literature already presents some corresponding observational equivalences. As hinted at in the introduction, however, they are *partial* answers since they all extend the λ-calculus with computational primitives, or embed it into process algebras:

- Dezani-Ciancaglini et al. [1998] add numerals, tests, and a non-deterministic choice operator, and show that $\mathcal{B}$ corresponds to *may convergence to a natural number* in their calculus.
- In the slightly different setting of *weak* head reduction, Sangiorgi [1994] shows that Lévy-Longo tree equality (the weak variant of $\mathcal{B}$) corresponds to various contextual equivalences all obtained via extended settings, namely the π-calculus, λ-calculus with non-determinism, and λ-calculus with well-formed operators. Similarly, Dezani-Ciancaglini et al. use another non-deterministic λ-calculus [1999], and Boudol and Laneve [1996] use a λ-calculus with modified syntax and rewriting rules where—crucially—terms can get stuck.
- Recently, Sakayori and Sangiorgi [2023] provide a characterization of $\mathcal{B}$ via the π-calculus.

In this paper, we do modify the λ-calculus via the checkers calculus, but we do *not* add extra features, we only refine the analysis of β-reduction in a quantitative and interactive way.

Clocked λ-Calculus. The idea of discriminating λ-terms by counting the head reduction steps in the construction of their Böhm trees also underlies the *clocked λ-calculus* of Endrullis et al. [2014, 2017]. The analogy ends there, as the clocked λ-calculus does not allow one to separate the internal/silent steps of terms and contexts from the external/interaction steps between the two.

Game Semantics. Our notion of interaction is inspired by the one between Player and Opponent in game semantics [Abramsky et al. 2000; Hyland and Ong 2000]—see Clairambault [2024] for the state of the art—in particular, silent steps are inspired by the hiding mechanism for composing games. We end up, however, with what seems to be a different setting. While the study of the relationship is left to future work, our interaction seems more basic (no need of the technical apparatus of game semantics), perfectly symmetrical, and allows for arbitrary shufflings of the two players in checkers terms, rather than keeping them apart as in game semantics. Moreover, with a few exceptions—namely, [Ker et al. 2003; Ong and Di Gianantonio 2002]—game models usually validate η-equivalence, while interaction equivalence does not. Notably, the game model by Ker et al. [2003] captures exactly $\mathcal{B}$.

Multi Types. The quantitative analysis of the relational semantics via multi types was pioneered by de Carvalho [2018]. Then Bernadet and Lengrand [2011]; de Carvalho et al. [2011] and especially Accattoli et al. [2020] developed and extended that approach, that has been applied to a variety of settings, including classical logic [Kesner and Vial 2020], the probabilistic λ-calculus [Dal Lago et al. 2021], reasonable space [Accattoli et al. 2022], and Bayesian networks [Faggian et al. 2024].

Variants of Relational Semantics. Relational semantics is generalized along several directions by Grellois and Melliès [2015]; Laird et al. [2013]; Ong [2017]. Laird et al.'s generalization of relations $r \subseteq A \times B$ from $r : A \times B \to \text{Bool}$ to $r : A \times B \to \mathcal{R}$ [2013], where $\mathcal{R}$ is a continuous semi-ring, might be used—in principle—to count interaction steps. The exact meaning of the coefficients in the interpretation of programs, however, is well-understood only at ground types, and remains unclear in the untyped case. Any temptation of seeing our checkers type system as a dichromatic version of Grellois and Melliès's *colored linear logic* [2015] should be avoided: in their work, colors need to correctly "match" in a type derivation because they represent different levels of priority.

Cost-Aware Denotational Semantics. Another line of research aims at defining cost-sensitive denotational semantics based on *sized* domains [Danner and Licata 2022; Kavvos et al. 2020]. These models are used to interpret a syntactic recurrence extracted from a given program, and prove a bounding theorem about such extraction. Niu and Harper [2023] propose a synthetic language for cost-aware denotational semantics, endowed with phase-separated constructions of intensional and extensional computations. These approaches are designed for typed languages with constants, where observations are made at ground type, but they are hardly generalizable to the untyped case.

There also are cost-aware game models such as Ghica's slot games [2005]—which capture Sands' improvements—and Alcolei et al.'s resource-tracking concurrent games [2019]. The relationship between these concurrency-driven models and our approach is deferred to future investigation.

10 Future Work

Weak Head. Our study focuses on the paradigmatic case of head reduction, but it could be adapted to weak head reduction (sometimes called *lazy* reduction [Abramsky and Ong 1993]). It is folklore that the Lévy-Longo tree preorder (a weak variant of Böhm trees) matches the weak head type preorder. We conjecture that the Lévy-Longo tree preorder matches exactly the weak head variant of our interaction preorder. Most of our study would adapt smoothly, but for the interaction Böhm out. The culprit is that there is no analogous of Böhm separation theorem for weak head reduction. There exist separation results but they all involve extensions of the λ-calculus (see above among related works). Crafting a weak interaction Böhm out might require some work.

Call-by-Value. In call-by-name, the gap between head normal form bisimilarity (aka interaction equivalence), and contextual equivalence is, roughly, extensionality. In call-by-value, the gap contains more computational principles, as stressed in particular by the recent works [Accattoli

et al. 2023; Accattoli and Lancelot 2024]. It would be interesting to adapt interaction equivalence to call-by-value and explore whether variants of the definition catch theories in between.

Back to Improvements. Now that there is a notion of equational improvement, it is natural to adapt it to call-by-need and compare / revisit / extend the results about Sands' (non-equational) improvements and program transformations in the literature [Gustavsson and Sands 2001; Hackett and Hutton 2014, 2015, 2018; Moran and Sands 1999; Sands 1996a; Schmidt-Schauß et al. 2018].

Game Semantics. To relate interaction equivalence with game semantics, it is natural to look at the Böhm tree game model by Ker et al. [2003]. Operational game semantics [Jaber and Sangiorgi 2022; Laird 2007; Levy and Staton 2014], a sub-area of game semantics based on labeled transition systems, is another natural candidate. More generally, our work provides a good justification for a systematic exploration of non-extensional game semantics.

PCF. Game semantics was introduced to capture denotationally PCF contextual equivalence, which is obtained via a quotient on games [Abramsky et al. 2000; Hyland and Ong 2000]. Is there a relationship between game models *before the quotient* and PCF interaction equivalence?

Higher-Order Model Checking. Higher-order model checking is strongly connected to game semantics [Ong 2006], intersection/multi types [Kobayashi 2009], and Böhm trees [Clairambault and Murawski 2013]. It is reasonable to expect a connection with interaction equivalence.

Complete Normal Form Bisimilarities. In some λ-calculi with effects, normal form bisimilarities are complete for contextual equivalences [Biernacki et al. 2019; Støvring and Lassen 2009]. We conjecture that, therein, contextual equivalence and interaction equivalence coincide.

Interaction Cost. It is natural to wonder what kind of interaction cost emerges from our study and how it relates to both the actual cost of computation and interaction (in)equivalence. The characterization of our interaction improvement in terms of Böhm trees reveals that one can improve the interactive cost of a λ-term by performing η-reductions, or by replacing some head-diverging sub-term—that is, an idle looping execution branch—with a non-looping one. Beyond Böhm tree characterizations, this may also relate to the length of interaction sequences in game semantics for the untyped call-by-name λ-calculus, where interaction improvements could reflect optimizations in communication between player and opponent.

We believe that our interaction semantics can also effectively capture communication costs in the evaluation of functional programs within distributed environments. In this framework, black and white terms correspond to processes running on different machines, with interaction steps modeling inter-process communication, while locally executable steps remain silent. To formalize this idea, we shall first adapt our interaction relations to a process calculus, and then investigate how the resulting improvements in interaction relates with measures of communication complexity.

11 Conclusions

Our work stems from the recognition of a tension between the equational aspect of contextual equivalences and the desire to observe the time cost of programs, expressed as the number of evaluation steps. We solve the tension by introducing the *checkers calculus*, a λ-calculus where the internal and external aspects of computation receive a first-class status. The new setting is then used to define *interaction equivalence*, which induces an equational theory for the ordinary λ-calculus: a very well-known one, namely the equality $\mathcal{B}$ of Böhm trees without η.

Beyond the technical aspects, the main takeaway is probably the framework, which is considerably simpler than other theories of interaction such as game semantics or the geometry of interaction, and not *ad-hoc*, as witnessed by the relationship with $\mathcal{B}$ and with multi types.

Acknowledgments

The authors would like to thank Guilhem Jaber for many discussions about operational game semantics for the λ-calculus, which eventually led us to the idea of tagging terms to distinguish between silent and interaction steps.

References

Samson Abramsky, Radha Jagadeesan, and Pasquale Malacaria. 2000. Full Abstraction for PCF. *Inf. Comput.* 163, 2 (2000), 409–470. https://doi.org/10.1006/INCO.2000.2930

Samson Abramsky and C.-H. Luke Ong. 1993. Full Abstraction in the Lazy Lambda Calculus. *Inf. Comput.* 105, 2 (1993), 159–267. https://doi.org/10.1006/INCO.1993.1044

Beniamino Accattoli, Ugo Dal Lago, and Gabriele Vanoni. 2022. Multi types and reasonable space. *Proc. ACM Program. Lang.* 6, ICFP (2022), 799–825. https://doi.org/10.1145/3547650

Beniamino Accattoli, Claudia Faggian, and Adrienne Lancelot. 2023. Normal Form Bisimulations By Value. *CoRR* abs/2303.08161 (2023). https://doi.org/10.48550/ARXIV.2303.08161 arXiv:2303.08161

Beniamino Accattoli, Stéphane Graham-Lengrand, and Delia Kesner. 2020. Tight typings and split bounds, fully developed. *J. Funct. Program.* 30 (2020), e14. https://doi.org/10.1017/S095679682000012X

Beniamino Accattoli and Adrienne Lancelot. 2024. Mirroring Call-By-Need, or Values Acting Silly. In *9th International Conference on Formal Structures for Computation and Deduction, FSCD 2024, July 10-13, 2024, Tallinn, Estonia (LIPIcs, Vol. 299)*, Jakob Rehof (Ed.). Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 23:1–23:24. https://doi.org/10.4230/LIPICS.FSCD.2024.23

Beniamino Accattoli, Adrienne Lancelot, Giulio Manzonetto, and Gabriele Vanoni. 2024. Interaction Equivalence. *CoRR* abs/2409.18709 (2024). https://doi.org/10.48550/ARXIV.2409.18709 arXiv:2409.18709

Peter Aczel. 1978. *A General Church-Rosser Theorem.* Technical Report. University of Manchester.

Aurore Alcolei, Pierre Clairambault, and Olivier Laurent. 2019. Resource-Tracking Concurrent Games. In *Foundations of Software Science and Computation Structures - 22nd International Conference, FOSSACS 2019, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2019, Prague, Czech Republic, April 6-11, 2019, Proceedings (Lecture Notes in Computer Science, Vol. 11425)*, Mikolaj Bojanczyk and Alex Simpson (Eds.). Springer, 27–44. https://doi.org/10.1007/978-3-030-17127-8_2

Henk Barendregt. 1977. The Type Free Lambda Calculus. In *Handbook of Mathematical Logic*, Jon Barwise (Ed.). Studies in Logic and the Foundations of Mathematics, Vol. 90. Elsevier, 1091 – 1132.

Henk Barendregt. 1984. *The Lambda Calculus – Its Syntax and Semantics.* Studies in logic and the foundations of mathematics, Vol. 103. North-Holland.

Henk Barendregt and Giulio Manzonetto. 2022. *A Lambda Calculus Satellite.* College Publications. https://www.collegepublications.co.uk/logic/mlf/?00035

Alexis Bernadet and Stéphane Lengrand. 2011. Complexity of Strongly Normalising λ-Terms via Non-idempotent Intersection Types. In *Foundations of Software Science and Computational Structures - 14th International Conference, FOSSACS 2011, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2011, Saarbrücken, Germany, March 26-April 3, 2011. Proceedings (Lecture Notes in Computer Science, Vol. 6604)*, Martin Hofmann (Ed.). Springer, 88–107. https://doi.org/10.1007/978-3-642-19805-2_7

Dariusz Biernacki, Sergueï Lenglet, and Piotr Polesiuk. 2019. A Complete Normal-Form Bisimilarity for State. In *Foundations of Software Science and Computation Structures - 22nd International Conference, FOSSACS 2019, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2019, Prague, Czech Republic, April 6-11, 2019, Proceedings (Lecture Notes in Computer Science, Vol. 11425)*, Mikolaj Bojanczyk and Alex Simpson (Eds.). Springer, 98–114. https://doi.org/10.1007/978-3-030-17127-8_6

Corrado Böhm. 1968. Alcune proprietà delle forme β-η-normali nel λ-K-calcolo. *Pubblicazioni dell'istituto per le applicazioni del calcolo* 696 (1968), 1–19. Lavoro eseguito all'INAC.

Gérard Boudol and Cosimo Laneve. 1996. The Discriminating Power of Multiplicities in the λ-Calculus. *Information and Computation* 126, 1 (1996), 83–102. https://doi.org/10.1006/inco.1996.0037

Flavien Breuvart, Giulio Manzonetto, and Domenico Ruoppolo. 2018. Relational Graph Models at Work. *Log. Methods Comput. Sci.* 14, 3 (2018). https://doi.org/10.23638/LMCS-14(3:2)2018

Antonio Bucciarelli, Delia Kesner, and Daniel Ventura. 2017. Non-idempotent intersection types for the Lambda-Calculus. *Log. J. IGPL* 25, 4 (2017), 431–464. https://doi.org/10.1093/JIGPAL/JZX018

Pierre Clairambault. 2024. *Causal Investigations in Interactive Semantics.* https://tel.archives-ouvertes.fr/tel-04523273

Pierre Clairambault and Andrzej S. Murawski. 2013. Böhm Trees as Higher-Order Recursive Schemes. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2013, December 12-14, 2013, Guwahati, India (LIPIcs, Vol. 24)*, Anil Seth and Nisheeth K. Vishnoi (Eds.). Schloss Dagstuhl - Leibniz-Zentrum für

Informatik, 91–102. https://doi.org/10.4230/LIPICS.FSTTCS.2013.91

Ugo Dal Lago, Claudia Faggian, and Simona Ronchi Della Rocca. 2021. Intersection types and (positive) almost-sure termination. *Proc. ACM Program. Lang.* 5, POPL (2021), 1–32. https://doi.org/10.1145/3434313

Norman Danner and Daniel R. Licata. 2022. Denotational semantics as a foundation for cost recurrence extraction for functional languages. *J. Funct. Program.* 32 (2022), e8. https://doi.org/10.1017/S095679682200003X

Daniel de Carvalho. 2007. *Sémantiques de la logique linéaire et temps de calcul.* Thèse de Doctorat. Université Aix-Marseille II.

Daniel de Carvalho. 2018. Execution time of λ-terms via denotational semantics and intersection types. *Math. Struct. Comput. Sci.* 28, 7 (2018), 1169–1203. https://doi.org/10.1017/S0960129516000396

Daniel de Carvalho, Michele Pagani, and Lorenzo Tortora de Falco. 2011. A semantic measure of the execution time in linear logic. *Theor. Comput. Sci.* 412, 20 (2011), 1884–1902. https://doi.org/10.1016/J.TCS.2010.12.017

Mariangiola Dezani-Ciancaglini. 2001. TLCA list, Problem #18: Find trees representing contextual equivalences. See http://tlca.di.unito.it/opltlca/.

Mariangiola Dezani-Ciancaglini, Benedetto Intrigila, and Marisa Venturini-Zilli. 1998. Böhm's theorem for Böhm trees. In *ICTCS*, Vol. 98. World Scientific, 1–23.

Mariangiola Dezani-Ciancaglini, Jerzy Tiuryn, and Pawel Urzyczyn. 1999. Discrimination by Parallel Observers: The Algorithm. *Information and Computation* 150, 2 (1999), 153–186. https://doi.org/10.1006/inco.1998.2773

Jörg Endrullis, Dimitri Hendriks, Jan Willem Klop, and Andrew Polonsky. 2014. Discriminating Lambda-Terms Using Clocked Boehm Trees. *Logical Methods in Computer Science* 10, 2 (2014). https://doi.org/10.2168/LMCS-10(2:4)2014

Jörg Endrullis, Dimitri Hendriks, Jan Willem Klop, and Andrew Polonsky. 2017. Clocked lambda calculus. *Math. Struct. Comput. Sci.* 27, 5 (2017), 782–806. https://doi.org/10.1017/S0960129515000389

Claudia Faggian, Daniele Pautasso, and Gabriele Vanoni. 2024. Higher Order Bayesian Networks, Exactly. *Proc. ACM Program. Lang.* 8, POPL (2024), 2514–2546. https://doi.org/10.1145/3632926

Dan R. Ghica. 2005. Slot games: a quantitative model of computation. In *Proceedings of the 32nd ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2005, Long Beach, California, USA, January 12-14, 2005*, Jens Palsberg and Martín Abadi (Eds.). ACM, 85–97. https://doi.org/10.1145/1040305.1040313

Charles Grellois and Paul-André Melliès. 2015. Relational Semantics of Linear Logic and Higher-order Model Checking. In *24th EACSL Annual Conference on Computer Science Logic, CSL 2015, September 7-10, 2015, Berlin, Germany (LIPIcs, Vol. 41)*, Stephan Kreutzer (Ed.). Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 260–276. https://doi.org/10.4230/LIPICS.CSL.2015.260

Jörgen Gustavsson and David Sands. 2001. Possibilities and Limitations of Call-by-Need Space Improvement. In *Proceedings of the Sixth ACM SIGPLAN International Conference on Functional Programming (ICFP '01), Firenze (Florence), Italy, September 3-5, 2001*, Benjamin C. Pierce (Ed.). ACM, 265–276. https://doi.org/10.1145/507635.507667

Jennifer Hackett and Graham Hutton. 2014. Worker/wrapper/makes it/faster. In *Proceedings of the 19th ACM SIGPLAN international conference on Functional programming, Gothenburg, Sweden, September 1-3, 2014*, Johan Jeuring and Manuel M. T. Chakravarty (Eds.). ACM, 95–107. https://doi.org/10.1145/2628136.2628142

Jennifer Hackett and Graham Hutton. 2015. Programs for Cheap!. In *30th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2015, Kyoto, Japan, July 6-10, 2015*. IEEE Computer Society, 115–126. https://doi.org/10.1109/LICS.2015.21

Jennifer Hackett and Graham Hutton. 2018. Parametric polymorphism and operational improvement. *Proc. ACM Program. Lang.* 2, ICFP (2018), 68:1–68:24. https://doi.org/10.1145/3236763

Martin Hyland. 1975. A survey of some useful partial order relations on terms of the lambda calculus. In *Lambda-Calculus and Computer Science Theory, Proceedings of the Symposium Held in Rome, Italy, March 25-27, 1975 (Lecture Notes in Computer Science, Vol. 37)*, Corrado Böhm (Ed.). Springer, 83–95. https://doi.org/10.1007/BFB0029520

Martin Hyland. 1976. A Syntactic Characterization of the Equality in Some Models for the Lambda Calculus. *Journal of the London Mathematical Society* s2-12, 3 (1976), 361–370. https://doi.org/10.1112/jlms/s2-12.3.361

Martin Hyland, Misao Nagayama, John Power, and Giuseppe Rosolini. 2004. A Category Theoretic Formulation for Engeler-style Models of the Untyped lambda. 161 (2004), 43–57. https://doi.org/10.1016/J.ENTCS.2006.04.024

Martin Hyland and C.-H. Luke Ong. 2000. On Full Abstraction for PCF: I, II, and III. *Inf. Comput.* 163, 2 (2000), 285–408. https://doi.org/10.1006/INCO.2000.2917

Benedetto Intrigila, Giulio Manzonetto, and Andrew Polonsky. 2019. Degrees of extensionality in the theory of Böhm trees and Sallé's conjecture. *Log. Methods Comput. Sci.* 15, 1 (2019). https://doi.org/10.23638/LMCS-15(1:6)2019

Guilhem Jaber and Davide Sangiorgi. 2022. Games, Mobile Processes, and Functions. In *30th EACSL Annual Conference on Computer Science Logic (CSL 2022) (Leibniz International Proceedings in Informatics (LIPIcs), Vol. 216)*, Florin Manea and Alex Simpson (Eds.). Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany, 25:1–25:18. https://doi.org/10.4230/LIPIcs.CSL.2022.25

G. A. Kavvos, Edward Morehouse, Daniel R. Licata, and Norman Danner. 2020. Recurrence extraction for functional programs through call-by-push-value. *Proc. ACM Program. Lang.* 4, POPL (2020), 15:1–15:31. https://doi.org/10.1145/3371083

Andrew D. Ker, Hanno Nickau, and C.-H. Luke Ong. 2003. Adapting innocent game models for the Böhm tree λ-theory. *Theor. Comput. Sci.* 308, 1-3 (2003), 333–366. https://doi.org/10.1016/S0304-3975(02)00849-6

Delia Kesner and Pierre Vial. 2020. Consuming and Persistent Types for Classical Logic. In *LICS '20: 35th Annual ACM/IEEE Symposium on Logic in Computer Science, Saarbrücken, Germany, July 8-11, 2020*, Holger Hermanns, Lijun Zhang, Naoki Kobayashi, and Dale Miller (Eds.). ACM, 619–632. https://doi.org/10.1145/3373718.3394774

Jan Willem Klop. 1980. *Combinatory Reduction Systems.* PhD thesis. Utrecht University.

Naoki Kobayashi. 2009. Types and higher-order recursion schemes for verification of higher-order programs. In *Proceedings of the 36th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2009, Savannah, GA, USA, January 21-23, 2009*, Zhong Shao and Benjamin C. Pierce (Eds.). ACM, 416–428. https://doi.org/10.1145/1480881.1480933

Jim Laird. 2007. A Fully Abstract Trace Semantics for General References. In *Automata, Languages and Programming, 34th International Colloquium, ICALP 2007, Wroclaw, Poland, July 9-13, 2007, Proceedings (Lecture Notes in Computer Science, Vol. 4596)*, Lars Arge, Christian Cachin, Tomasz Jurdzinski, and Andrzej Tarlecki (Eds.). Springer, 667–679. https://doi.org/10.1007/978-3-540-73420-8_58

Jim Laird, Giulio Manzonetto, Guy McCusker, and Michele Pagani. 2013. Weighted Relational Models of Typed Lambda-Calculi. In *28th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2013, New Orleans, LA, USA, June 25-28, 2013*. IEEE Computer Society, 301–310. https://doi.org/10.1109/LICS.2013.36

Søren B. Lassen. 1999. Bisimulation in Untyped Lambda Calculus: Böhm Trees and Bisimulation up to Context. 20 (1999), 346–374. https://doi.org/10.1016/S1571-0661(04)80083-5

Paul Blain Levy and Sam Staton. 2014. Transition systems over games. In *Proceedings of the 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)* (Vienna, Austria) *(CSL-LICS '14)*. Association for Computing Machinery, New York, NY, USA, Article 64, 10 pages. https://doi.org/10.1145/2603088.2603150

Andrew Moran and David Sands. 1999. Improvement in a Lazy Context: An Operational Theory for Call-by-Need. In *POPL '99, Proceedings of the 26th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, San Antonio, TX, USA, January 20-22, 1999*, Andrew W. Appel and Alex Aiken (Eds.). ACM, 43–56. https://doi.org/10.1145/292540.292547

James Hiram Morris. 1968. *Lambda-calculus Models of Programming Languages.* Ph. D. Dissertation. Massachusetts Institute of Technology. https://books.google.is/books?id=DklAAQAAIAAJ

Koko Muroya and Makoto Hamana. 2024. Term Evaluation Systems with Refinements: First-Order, Second-Order, and Contextual Improvement. In *Functional and Logic Programming - 17th International Symposium, FLOPS 2024, Kumamoto, Japan, May 15-17, 2024, Proceedings (Lecture Notes in Computer Science, Vol. 14659)*, Jeremy Gibbons and Dale Miller (Eds.). Springer, 31–61. https://doi.org/10.1007/978-981-97-2300-3_3

Tobias Nipkow. 1991. Higher-Order Critical Pairs. In *Proceedings of the Sixth Annual Symposium on Logic in Computer Science (LICS '91), Amsterdam, The Netherlands, July 15-18, 1991*. IEEE Computer Society, 342–349. https://doi.org/10.1109/LICS.1991.151658

Yue Niu and Robert Harper. 2023. A Metalanguage for Cost-Aware Denotational Semantics. In *38th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2023, Boston, MA, USA, June 26-29, 2023*. IEEE, 1–14. https://doi.org/10.1109/LICS56636.2023.10175777

C.-H. Luke Ong. 2006. 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*. IEEE Computer Society, 81–90. https://doi.org/10.1109/LICS.2006.38

C.-H. Luke Ong. 2017. Quantitative semantics of the lambda calculus: Some generalisations of the relational model. In *32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017, Reykjavik, Iceland, June 20-23, 2017*. IEEE Computer Society, 1–12. https://doi.org/10.1109/LICS.2017.8005064

C.-H. Luke Ong and Pietro Di Gianantonio. 2002. Games Characterizing Lévy-Longo Trees. In *Automata, Languages and Programming, 29th International Colloquium, ICALP 2002, Malaga, Spain, July 8-13, 2002, Proceedings (Lecture Notes in Computer Science, Vol. 2380)*, Peter Widmayer, Francisco Triguero Ruiz, Rafael Morales Bueno, Matthew Hennessy, Stephan J. Eidenbenz, and Ricardo Conejo (Eds.). Springer, 476–487. https://doi.org/10.1007/3-540-45465-9_41

Marco Patrignani, Amal Ahmed, and Dave Clarke. 2019. Formal Approaches to Secure Compilation: A Survey of Fully Abstract Compilation and Related Work. *ACM Comput. Surv.* 51, 6 (2019), 125:1–125:36. https://doi.org/10.1145/3280984

Gordon D. Plotkin. 1977. LCF Considered as a Programming Language. *Theor. Comput. Sci.* 5, 3 (1977), 223–255. https://doi.org/10.1016/0304-3975(77)90044-5

James Riely and Jan F. Prins. 2000. Flattening Is an Improvement. In *Static Analysis, 7th International Symposium, SAS 2000, Santa Barbara, CA, USA, June 29 - July 1, 2000, Proceedings (Lecture Notes in Computer Science, Vol. 1824)*, Jens Palsberg (Ed.). Springer, 360–376. https://doi.org/10.1007/978-3-540-45099-3_19

Ken Sakayori and Davide Sangiorgi. 2023. Extensional and Non-extensional Functions as Processes. In *38th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2023, Boston, MA, USA, June 26-29, 2023*. IEEE, 1–13. https://doi.org/10.1109/LICS56636.2023.10175686

David Sands. 1996a. Proving the Correctness of Recursion-Based Automatic Program Transformations. *Theor. Comput. Sci.* 167, 1&2 (1996), 193–233. https://doi.org/10.1016/0304-3975(96)00074-6

David Sands. 1996b. Total correctness by local improvement in the transformation of functional programs. *ACM Trans. Program. Lang. Syst.* 18, 2 (mar 1996), 175–234. https://doi.org/10.1145/227699.227716

David Sands. 1999. *Improvement theory and its applications.* Cambridge University Press, USA, 275–306.

Davide Sangiorgi. 1994. The Lazy Lambda Calculus in a Concurrency Scenario. *Inf. Comput.* 111, 1 (1994), 120–153. https://doi.org/10.1006/INCO.1994.1042

Manfred Schmidt-Schauß, David Sabel, and Nils Dallmeyer. 2018. Sequential and Parallel Improvements in a Concurrent Functional Programming Language. In *Proceedings of the 20th International Symposium on Principles and Practice of Declarative Programming, PPDP 2018, Frankfurt am Main, Germany, September 03-05, 2018*, David Sabel and Peter Thiemann (Eds.). ACM, 20:1–20:13. https://doi.org/10.1145/3236950.3236952

Dana S. Scott. 1972. Continuous lattices. In *Toposes, Algebraic Geometry and Logic (Lecture Notes in Mathematics, Vol. 274)*, Lawvere (Ed.). Springer, 97–136. https://doi.org/10.1007/BFb0073967

Kristian Støvring and Søren B. Lassen. 2009. A Complete, Co-inductive Syntactic Theory of Sequential Control and State. In *Semantics and Algebraic Specification, Essays Dedicated to Peter D. Mosses on the Occasion of His 60th Birthday (Lecture Notes in Computer Science, Vol. 5700)*, Jens Palsberg (Ed.). Springer, 329–375. https://doi.org/10.1007/978-3-642-04164-8_17

Christopher P. Wadsworth. 1976. The Relation Between Computational and Denotational Properties for Scott's $\mathcal{D}_\infty$-Models of the Lambda-Calculus. *SIAM J. Comput.* 5, 3 (1976), 488–521. https://doi.org/10.1137/0205036

Received 2024-07-11; accepted 2024-11-07