Semantic Bounds and Multi Types, Revisited

Beniamino Accattoli
Inria & LIX, École Poytechnique, Palaiseau, France

Abstract

Intersection types are a standard tool in operational and semantical studies of the λ-calculus. De Carvalho showed how multi types, a quantitative variant of intersection types providing a handy presentation of the relational denotational model, allows one to extract precise bounds on the number of β-steps and the size of normal forms.

In the last few years, de Carvalho's work has been extended and adapted to a number of λ-calculi, evaluation strategies, and abstract machines. These works, however, only adapt the first part of his work, that extracts bounds from multi type *derivations*, while never consider the second part, which deals with extracting bounds from the multi types themselves. The reason is that this second part is more technical, and requires to reason up to *type substitutions*. It is however also the most interesting, because it shows that the bounding power is *inherent* to the relational model (which is induced by the types, without the derivations), independently of its presentation as a type system.

Here we dissect and clarify the second part of de Carvalho's work, establishing a link with principal multi types, and isolating a key property independent of type substitutions.

2012 ACM Subject Classification Theory of computation → Lambda calculus; Theory of computation → Denotational semantics; Theory of computation → Operational semantics

Keywords and phrases Lambda calculus, intersection types, denotational semantics, linear logic

Digital Object Identifier 10.4230/LIPIcs.CSL.2024.7

Related Version *Paper with proof appendix hosted on arXiv*: `http://arxiv.org/abs/2311.18233` [1]

1 Introduction

Denotational semantics studies invariants of program evaluation. The typical way in which it is connected to the operational semantics of λ-calculi is at the *qualitative* level, via *adequacy*: the denotational interpretation $[\![t]\!]$ of a λ-term t is non-trivial (typically non-empty) if and only if the evaluation of t terminates.

At first sight, denotational semantics cannot provide *quantitative* operational insights such as evaluation lengths, because of its invariance by evaluation. Things are in fact not so black and white. Being invariant by evaluation, denotational semantics models normal forms, and in a *compositional* way: by composing the interpretations of two terms one can obtain the interpretation of the result of their application – therefore, denotational semantics does reflect the evaluation process *somehow*.

The aim of this paper is to revisit some overlooked – but we believe important – results for the λ-calculus by de Carvalho, about the extraction of bounds on evaluation lengths and the size of normal forms from the interpretation of terms into the relational model.

Relational Semantics and Multi Types. The relational model [37, 17] is a simple denotational semantics of the λ-calculus induced by the relational model of linear logic, via the representation of the λ-calculus in linear logic. It is a paradigmatic model, underlying many others [30, 31, 24, 45, 48], mainly studied by Ehrhard and his students and co-authors [38, 18, 21, 16, 32, 33, 34, 26, 47, 46, 49, 15], the importance of which has emerged slowly. One of its features is that it admits a handy presentation via an intersection type system.

The distinguished property of intersection types is that they *characterize* termination properties, in the sense that they not only enforce termination, but they also type *all* terminating terms. Additionally, by tuning the definition of the type system, one can capture

different notions of termination (weak head/head/leftmost termination, strong normalization, call-by-value/need, etc). Such a characterization usually induces a model: the set of types for a term t is an invariant of the characterized notion of evaluation $\to$, which gives rise to a denotational semantics which is adequate for $\to$. Therefore, intersection type systems are a tool halfway the operational and the denotational semantics of the λ-calculus.

Multi types are a *linear logic-related* variant of intersection types where intersections are *non-idempotent* (they are also known as *non-idempotent intersection types*), that is where $A \cap A \neq A$. The set of multi types judgements derivable for a term t provides a denotation $[\![t]\!]$ which coincides with the interpretation into the relational model.

Multi Types and Quantitative Bounds. In a seminal work, de Carvalho used the multi type system to obtain exact bounds on the evaluation lengths and the size of normal form for λ-terms [25, 27]. The relevance of his work was fully appreciated by the community only a decade later (as surveyed below), when it blossomed into a number of variations and extensions by other authors. De Carvalho developed *two* main results, but only the first one has widely permeated the community. The second one is arguably the most technical but also the deeper one. The aim of this paper is to make it accessible to a wider audience.

De Carvalho's original presentation in [25, 27] uses multi types to measure two forms of strong evaluations realized by abstract machines, implementing the head and leftmost call-by-name strategies. In this overview, we prefer to slightly depart from the (by now somewhat outdated) details of his work, forgetting about abstract machines, focussing on leftmost evaluation (the head case is treated at the end of the paper), and isolating three (rather than two) kinds of bounds:

1. *Bounds from type derivations.* The size $|\Phi|$ of a type derivation $\Phi \rhd \Gamma \vdash t : \mathsf{L}$ bounds the number of leftmost steps from t to its normal form $\mathtt{nf}(t)$ *plus* the size $|\mathtt{nf}(t)|$ of the normal form. Moreover, derivations of minimal size provide exact bounds.
2. *Size bounds from types.* The types in the final judgment – a point of the relational interpretation $[\![t]\!]$ – also provide a bound, independently of the derivation Φ. Being invariant by evaluation, they cannot bound evaluation lengths. They do however bound the size $|\mathtt{nf}(t)|$ of the normal form, and bounds are exact when types are minimal.
3. *Bounds from composable types.* De Carvalho shows that types can be used to bound evaluation lengths *compositionally*: from the types in $[\![t]\!]$ and $[\![u]\!]$ it is possible to extract bounds about the leftmost evaluation and the normal form of tu, with *no reference to type derivations*. Exact bounds rest on a complex construction involving type substitutions.

The third kind of bounds is de Carvalho's second result, and it is where the bounding power of the multi type system (which is just one possible way of defining relational semantics) is *lifted* to relational semantics. Therefore, the lifting guarantees that the bounding power is an *inherent* feature of the relational model – multi types are just a handy tool to show it.

Of the three results, the third one is the most technical. In particular, it requires to enrich the type system with an infinity of type variables and work up to type substitutions. The puzzling fact is that these extra features play no role in the two previous points.

De Carvalho's Legacy. De Carvalho developed his results in his PhD, defended in 2007 [25]. His work was known by the community thanks to a technical report, turned into a journal paper only much later, in 2018 [27]. Soon after his PhD, he adapted his work to linear logic, with Pagani and Tortora de Falco [28, 29]. Then, Bernadet and Graham-Lengrand adapted his work to measure the longest evaluation in the λ-calculus [14], but they only provided bounds of the first kind (*from type derivations*).

At the time, it was not known whether it would make sense to count the number of β-steps (or linear logic cut-elimination steps) as a reasonable measure of complexity. After the case of the λ-calculus was clarified in the positive by Accattoli and Dal Lago [2], de Carvalho's work has been revisited by Accattoli et al. [7]. The revisitation started a new wave of works adapting de Carvalho's study to many evaluation strategies and extensions of the λ-calculus, including call-by-value [8, 9], call-by-need [10], a linear logic presentation of call-by-push-value [19, 42], the $\lambda\mu$-calculus [41], the λ-calculus with pattern matching [13], generalized applications [35], fully lazy sharing [39], global state [12], the probabilistic λ-calculus [23], the abstract machine underlying the geometry of interaction [3], and even adapted as to measure *space* [4, 5]. All these works provide bounds of the first kind, and some of them also of the second kind, but *none of them* deals with those of the third kind (*bounds from composable types*).

Contributions. This paper revisits the *bounds from composable types*, appeared only in [27, 28]. Beyond providing cleaner proofs of the results, we have a very close look at these bounds, isolating the subtleties and decomposing the proofs in smaller steps. In particular:

- *Subtlety 1, minimality does not work*: when bounding a single term, both derivations and types provide exact bounds when they are minimal. When dealing with the application of t to u, every pair of composable types for them provides bounds. We stress that the technicalities for bounds from composable types are inherent to the problem, since the minimal composable pair in general does *not* provide exact bounds.
- *Subtlety 2, the gap between derivations and types*: we draw attention to the fact that the previous subtlety stems from a fact about derivations in *isolation*, namely that, for a normal term t, both the derivation $\Phi \rhd \Gamma \vdash t : \mathsf{L}$ and the types in Γ and L provide bounds for $|t|$, but they may not coincide. In general, the bound from types is *laxer*. The bounds gap hinders the possibility of lifting the bounding power from derivations to types, if bounds from some derivations are not reflected by any type in the interpretation $[\![t]\!]$.
- *Dry representation and type substitutions*: we isolate the property behind de Carvalho's solution of this problem, which rests on type variables and type substitutions, and we connect it with the study of principal types. The idea is that given a type derivation $\Phi \rhd \Gamma \vdash t : \mathsf{L}$ for which there is a gap between the bound from Φ and the bound from Γ and L, there always is a second *dry* derivation $\Psi \rhd \Delta \vdash t : \mathsf{L}'$, whose types Δ and L' give the same bound as the first derivation Φ, *plus* a type substitution σ turning Ψ into Φ. Then, all bounds coming from derivations can also be seen as coming from types, potentially from the types of other derivations – the dry ones – but related via type substitutions.
- *Removing substitutions*: lastly, we show that *weaker* bounds from composable types can be obtained *without* dealing with an infinity of type variables and type substitutions. This point provides both a simplified route to the (slightly weaker) bounds and an explanation of why these additional technicalities are needed for the full result.

Internal *vs* External View. The problem studied in the paper can also be seen in a more abstract way. Terms, or more generally programs, can be studied from two perspectives, which are distinct and yet entangled. The internal view considers programs as closed systems and looks at their internal evaluation in isolation. In the external view, programs are seen as parts of larger systems. What is relevant is how programs compose and interact with each other, their internal evaluation is instead secondary and hidden.

Cost analyses are usually done following the internal view, while denotational semantics and types are external-oriented tools. Normal forms can be seen as internalizing the external information, as they are normal for the internal process and thus retain only information for potential external interactions. Multi type derivations capture both the number of steps of typed terms (which is an internal information) and the structure of their normal forms (which is the internalization of external information). Multi types instead capture the external information only (each type capturing a potential interaction).

Bounds from composable types connect internal and external information, as they use types (that are external) to bound the length of evaluation (which is internal) of the isolated system given by the two composed terms. The bounds are obtained building over the connection between types and normal forms. The difficulty in this study stems from the fact that in general there is a gap between how the external information is represented in types (richly, distinguishing between different interactions) and how it is internalized in normal forms (in a raw way, all possible interactions are flattened on a single object).

Limits of de Carvalho's Approach. The aim of this paper is also to highlight a fundamental limitation of de Carvalho's work. As we ourselves suggested, bounds from composable types can be seen as lifting the bounding power from the type system to the relational model. There is however a delicate point, as the lifting does not cover the whole of the model. The relational model, indeed, does not only interpret full normal forms and terminating terms, but also *head* normal forms and terms that are *head normalizable*. In particular, there are terms that are *hereditarily* head normalizing and yet never fully normalize, such as fix-point combinators. For these terms, which have non-empty interpretation in the relational model, de Carvalho's study does not say anything, because it assumes that the terms to be composed are fully normal. De Carvalho's bounds for head reduction do not solve the issue, they rather make it worse, since his theorems for the head case still have to assume that the composed terms are fully normal, which seems an unnatural assumption that is nonetheless mandatory.

Consequently, de Carvalho's technique does not apply to all terms having non-empty interpretation in the relational model. Here, we point out the problem, which is a first essential step. We do not aim at solving it, because it seems to require the development of a new approach, not just a refinement of de Carvalho's. Abstractly, the limits of his technique come from the fact that external information is measured by reducing it to internal information, rather than measuring it *separately* from it, which would allow one to measure external information even when the internal evaluation process does not fully terminate.

Proofs. A few proofs are omitted and can be found in the Proof Appendix available on ArXiv [1].

2 Head and Leftmost Reductions and Normal Forms

Basics of λ. The set Λ of untyped λ-terms is defined by $t ::= x \mid \lambda x.t \mid t\,t$, which are considered as quotiented by α-equivalence. The capture-avoiding substitution of x by u in t is written $t\{x \leftarrow u\}$.

β-Reduction and Normal Forms. β-reduction $\to_\beta \subseteq \Lambda \times \Lambda$ is defined as follows:

β-Reduction

$$\frac{}{(\lambda x.t)u \to_\beta t\{x \leftarrow u\}}\,ax \qquad \frac{t \to_\beta t'}{\lambda x.t \to_\beta \lambda x.t'}\,\lambda \qquad \frac{t \to_\beta t'}{tu \to_\beta t'u}\,@l \qquad \frac{t \to_\beta t'}{ut \to_\beta ut'}\,@r$$

It is well known that β-reduction is non-deterministic but confluent, that is, a term can have at most one normal form. Normal forms (for β) are described by the following grammar relying on the mutually inductive notion of neutral term:

$$\text{Neutral terms} \quad n ::= x \mid nf \qquad\qquad \text{Normal forms} \quad f ::= n \mid \lambda x.f$$

An alternative streamlined definition of normal forms is $f := \lambda x_1 \dots \lambda x_n.(y f_1 \dots f_k)$ with $n, k \geq 0$, y possibly equal to one of the x_i, and the terms f_j normal forms themselves.

For our quantitative study, we need a notion of size of normal forms. We use the following *inner* one, which counts the number of inner nodes of a term, when seen as a tree having variables as leaves, as it is the one that best matches what shall be measured via multi types.

▶ **Definition 1** (Inner size). *The inner size of λ-terms is defined as follows:*

$$|x|_{\mathtt{in}} := 0 \qquad |\lambda x.u|_{\mathtt{in}} := |u|_{\mathtt{in}} + 1 \qquad |ur|_{\mathtt{in}} := |u|_{\mathtt{in}} + |r|_{\mathtt{in}} + 1$$

Head Reduction. A deterministic notion of evaluation for λ-terms is *head reduction*, which reduces only the left branch of a term, when seen as a tree. The definition follows (it is obtained by omitting rule $@r$ in the definition of β, and constraining t not to be an abstraction in rule $@l$):

$$\text{Head reduction}$$

$$\frac{}{(\lambda x.t)u \to_{\mathtt{h}} t\{x \leftarrow u\}}\, ax \qquad \frac{t \to_{\mathtt{h}} t'}{\lambda x.t \ \to_{\mathtt{h}} \lambda x.t'}\, \lambda \qquad \frac{t \to_{\mathtt{h}} t' \quad t \neq \lambda x.r}{tu \to_{\mathtt{h}} t'u}\, @l$$

Let $\mathsf{I} := \lambda z.z$ be the identity combinator, $\delta := \lambda x.xx$ be the duplicator, and consider the following examples: $\delta\mathsf{I}t \to_{\mathtt{h}} \mathsf{I}\mathsf{I}t$ and $\lambda y.(\delta\mathsf{I}) \to_{\mathtt{h}} \lambda y.\mathsf{I}\mathsf{I}$ but $(\lambda y.(\delta\mathsf{I}))t \not\to_{\mathtt{h}} (\lambda y.\mathsf{I}\mathsf{I})t$, as it rather reduces to $(\delta\mathsf{I})\{y \leftarrow t\} = \delta\mathsf{I}$.

Head reduction might not compute normal forms, since it does not evaluate arguments. Its notion of normal form follows:

$$\text{Head normal forms} \quad h ::= \lambda x_1 \dots \lambda x_n.(y t_1 \dots t_k) \text{ with } n, k \geq 0 \text{ and } y \text{ possibly equal to one of the } x_i.$$

We shall also need a notion of *head size* for head normal forms defined as $|h|_{\mathtt{h}} := n + k$ if $h = \lambda x_1 \dots \lambda x_n.(y t_1 \dots t_k)$.

Leftmost Reduction. Leftmost-outermost reduction $\to_{\mathtt{lo}}$ (shortened to *leftmost*) is a deterministic extension of head reduction as to reduce arguments and reach normal forms. The definition relies on the notion of neutral term n used to describe normal forms.

$$\text{Leftmost(-Outermost) Reduction}$$

$$\frac{}{(\lambda x.t)u \to_{\mathtt{lo}} t\{x \leftarrow u\}}\, ax \qquad \frac{t \to_{\mathtt{lo}} t'}{\lambda x.t \ \to_{\mathtt{lo}} \lambda x.t'}\, \lambda$$

$$\frac{t \to_{\mathtt{lo}} t' \quad t \neq \lambda x.r}{tu \to_{\mathtt{lo}} t'u}\, @l \qquad \frac{n \text{ is neutral} \quad t \to_{\mathtt{lo}} t'}{nt \to_{\mathtt{lo}} nt'}\, @r$$

Examples: $x(\mathsf{I}\mathsf{I})(\mathsf{I}\mathsf{I}) \to_{\mathtt{lo}} x\mathsf{I}(\mathsf{I}\mathsf{I})$ but $x(\mathsf{I}\mathsf{I})(\mathsf{I}\mathsf{I}) \not\to_{\mathtt{lo}} x(\mathsf{I}\mathsf{I})\mathsf{I}$ and $\delta(\mathsf{I}\mathsf{I})(\mathsf{I}\mathsf{I}) \not\to_{\mathtt{lo}} \delta\mathsf{I}(\mathsf{I}\mathsf{I})$.

Leftmost normal forms are simply normal forms and – crucially – leftmost reduction is *normalizing*, that is, if t has a β-reduction $t \to_\beta^* f$ to normal form then leftmost reduction reaches that normal form, that is, $t \to_{\mathtt{lo}}^* f$. For a recent simple proof of this classic result see Accattoli et al. [6].

$$\frac{}{x:[\mathsf{L}]\vdash x:\mathsf{L}}\,\mathsf{ax} \qquad \frac{\Gamma\vdash t:\mathsf{L}}{\Gamma\setminus\!\setminus x\vdash \lambda x.t:\Gamma(x)\multimap\mathsf{L}}\,\lambda \qquad \frac{\Gamma\vdash t:\mathsf{M}\multimap\mathsf{L}\quad \Delta\vdash u:\mathsf{M}}{\Gamma\uplus\Delta\vdash tu:\mathsf{L}}\,@ \qquad \frac{[\Gamma_i\vdash t:\mathsf{L}_i]_{i\in I}}{\biguplus_{i\in I}\Gamma_i\vdash t:[\mathsf{L}_i]_{i\in I}}\,\mathsf{many}$$

Figure 1 De Carvalho's multi type system.

3 Multi Types, Head Reduction, and Bounds From Type Derivations

In this section, we give our presentation of de Carvalho's system of multi types, and recall some results from the literature. In particular, we recall the characterization of head reduction, how multi types induce the relational model, and the bounds that can be extracted from type derivations for the length of head evaluations and the head size of head normal forms.

Multi Types. There are two layers of types, *linear* and *multi types*, built over a countably infinite set of (linear) type variables:

$$\begin{array}{rrcl} \textsc{Linear type variables} & \mathtt{TyVars} & := & \{\mathsf{X},\mathsf{Y},\mathsf{Z},\mathsf{W},\mathsf{X}_1,\mathsf{Y}',\mathsf{Z}_2,\dots\} \\ \textsc{Linear types} & \mathsf{L},\mathsf{L}' & ::= & \mathsf{X}\in\mathtt{TyVars} \mid \mathsf{M}\multimap\mathsf{L} \\ \textsc{Multi types} & \mathsf{M},\mathsf{N} & ::= & [\mathsf{L}_1,\dots,\mathsf{L}_n] \quad n\in\mathbb{N} \\ \textsc{Generic types} & \mathsf{T},\mathsf{T}' & ::= & \mathsf{L}\mid\mathsf{M} \end{array}$$

where $[\mathsf{L}_1,\dots,\mathsf{L}_n]$ is our notation for finite multisets. The *empty* multi type $[\,]$ obtained by taking $n=0$ is also denoted by $\mathbf{0}$. Often, multi types are presented using a single type variable X instead of countably many. Most results are unaffected, but we shall see that for de Carvalho's semantic bounds we need countably many type variables.

A multi type $[\mathsf{L}_1,\dots,\mathsf{L}_n]$ has to be intended as a conjunction $\mathsf{L}_1\wedge\cdots\wedge\mathsf{L}_n$ of linear types $\mathsf{L}_1,\dots,\mathsf{L}_n$, for a commutative, associative, non-idempotent conjunction $\wedge$ (morally a tensor $\otimes$), of neutral element $\mathbf{0}$. The intuition is that a linear type corresponds to a single use of a term t, and that t is typed with a multiset M of n linear types if it is going to be used (at most) n times, that is, if t is part of a larger term u, then a copy of t shall end up in evaluation position during the evaluation of u.

Typing Rules. The derivation rules for the multi types system are in Figure 1. *Judgments* have shape $\Gamma\vdash t:\mathsf{M}$ or $\Gamma\vdash t:\mathsf{L}$ where t is a term, M is a multi type, L is a linear type, and Γ is a *type context*, that is, a total function from variables to multi types such that $\mathsf{dom}(\Gamma):=\{x\mid\Gamma(x)\neq\mathbf{0}\}$ is finite, usually represented as $x_1:\mathsf{M}_1,\dots,x_n:\mathsf{M}_n$ (for some $n\in\mathbb{N}$) if $\mathsf{dom}(\Gamma)\subseteq\{x_1,\dots,x_n\}$ and $\Gamma(x_i)=\mathsf{M}_i$ for all $1\leq i\leq n$.

The abstraction rule λ uses the notation $\Gamma\setminus\!\setminus x$ for the type context defined as Γ on every variable but possibly x, for which $(\Gamma\setminus\!\setminus x)(x)=\mathbf{0}$. It is a compact way to express the rule in both the cases $x\in\mathsf{dom}(\Gamma)$ and $x\notin\mathsf{dom}(\Gamma)$. Note that the application rule @ requires the argument to be typed with a multi type M, which is necessarily introduced by rule many, the hypotheses of which are a multi set of derivations, indexed by a possibly empty set I. When I is empty, the rule has no premises and can type every term. For instance, $\vdash\Omega:\mathbf{0}$ is derivable, but no linear type can be assigned to Ω. Essentially, $\mathbf{0}$ is the type of erasable terms, and every term is erasable in the λ-calculus.

Technicalities about Types. The type context Γ is *empty* if $\mathsf{dom}(\Gamma)=\emptyset$. *Multi-set sum* $\uplus$ is extended to type contexts point-wise, *i.e.* $(\Gamma\uplus\Delta)(x):=\Gamma(x)\uplus\Delta(x)$ for each variable x. This notion is extended to a finite family of type contexts as expected, in particular $\biguplus_{i\in J}\Gamma_i$ is the empty context when $J=\emptyset$. Given two type contexts Γ and Δ such that

$\mathsf{dom}(\Gamma) \cap \mathsf{dom}(\Delta) = \emptyset$, the type context Γ, Δ is defined by $(\Gamma, \Delta)(x) := \Gamma(x)$ if $x \in \mathsf{dom}(\Gamma)$, $(\Gamma, \Delta)(x) := \Delta(x)$ if $x \in \mathsf{dom}(\Delta)$, and $(\Gamma, \Delta)(x) := \mathbf{0}$ otherwise. Note that $\Gamma, x : \mathbf{0} = \Gamma$, where we implicitly assume $x \notin \mathsf{dom}(\Gamma)$.

Type Derivations. We write $\Phi \rhd \Gamma \vdash t : \mathsf{T}$ if Φ is a (*type*) *derivation* (*i.e.* a tree constructed using the rules in Figure 1) with conclusion the multi judgment $\Gamma \vdash t : \mathsf{T}$. In particular, we write $\Phi \rhd \vdash t : \mathsf{T}$ when Γ is empty. We write $\Phi \rhd t$ if $\Phi \rhd \Gamma \vdash t : \mathsf{T}$ for some type context Γ and some type T.

We need a notion of size of type derivations, which shall be used to extract bounds for the number of evaluation steps and the size of normal forms.

▶ **Definition 2** (Inner size of derivations). *Let Φ be a type derivation. The* inner size $|\Phi|_{\mathtt{in}}$ *of Φ is the number of occurrences of rules λ and @ in Φ.*

Subject Reduction and Expansion, and Relational Semantics. The first properties of the type system that we recall are subject reduction and expansion, which hold for *every* β-step.

▶ **Proposition 3.** *Let $t \to_\beta t'$.*

1. Subject reduction*: if $\Phi \rhd \Gamma \vdash t : \mathsf{L}$ then there is a derivation $\Phi' \rhd \Gamma \vdash t' : \mathsf{L}$ such that $|\Phi'|_{\mathtt{in}} \leq |\Phi|_{\mathtt{in}}$. Moreover, if $t \to_{\mathtt{h}} t'$ then $|\Phi'|_{\mathtt{in}} = |\Phi|_{\mathtt{in}} - 2$.*
2. Subject expansion*: if $\Phi' \rhd \Gamma \vdash t' : \mathsf{L}$ then there is a derivation $\Phi \rhd \Gamma \vdash t : \mathsf{L}$.*

Together, subject reduction and expansion state that type judgements are *invariants* of β-reduction. Such invariants actually induce a denotational model of the λ-calculus, its (call-by-name) *relational semantics*.

Let t be a term and $x_1, \dots, x_n$ ($n \geq 0$) be pairwise distinct variables. The list $\vec{x} = (x_1, \dots, x_n)$ is *suitable for* t if $\mathsf{fv}(t) \subseteq \{x_1, \dots, x_n\}$. If $\vec{x} = (x_1, \dots, x_n)$ is suitable for t, the *relational semantics* $[\![t]\!]_{\vec{x}}$ *of* t *for* $\vec{x}$ is defined by:

$$[\![t]\!]_{\vec{x}} \quad := \quad \{((\mathsf{M}_1, \dots, \mathsf{M}_n), \mathsf{L}) \mid \exists\, \Phi \rhd x_1 : \mathsf{M}_1, \dots, x_n : \mathsf{M}_n \vdash t : \mathsf{L}\}.$$

The following property is an immediate corollary of subject reduction and expansion.

▶ **Proposition 4** (Invariance). *Let $\vec{x} = (x_1, \dots, x_n)$ be suitable for two terms t and u. If $t \to_\beta u$ then $[\![t]\!]_{\vec{x}} = [\![u]\!]_{\vec{x}}$.*

Characterizing Head Termination. Note the quantitative aspect of subject reduction (Prop. 3.1), stating that the derivation size *cannot increase* after a reduction step, and that it *decreases* with head steps (of 2 because removing a head β redex removes a λ and a @ rule). It does not say that it decreases at *every* step because steps occurring in sub-terms typed with rule many might not change the size. For instance, if $xt \to xt'$ and t is typed using a empty many rule (*i.e.* with 0 premises), which is a sub-derivation of size 0, then also t' is typed using a empty many rule, of size 0. Therefore, not all typable terms terminate, as for instance $x\Omega$ is typable as follows, for any linear type L, but it has no normal form:

$$\dfrac{\dfrac{}{x : [[\mathbf{0}] \multimap \mathsf{L}] \vdash x : [\mathbf{0}] \multimap \mathsf{L}}\,\mathsf{ax} \qquad \dfrac{}{\vdash \Omega : \mathbf{0}}\,\mathsf{many}}{x : [[\mathbf{0}] \multimap \mathsf{L}] \vdash x\Omega : \mathsf{L}}\,@ \tag{1}$$

[0] More precisely, such a model is the restriction of the relational model for lineal logic to the image of Girard's [36] call-by-name translation $(A \Rightarrow B)^{\mathsf{n}} = !A^{\mathsf{n}} \multimap B^{\mathsf{n}}$ of the intuitionistic arrow into linear logic.

Since the size of type derivations decreases at every head step, it provides a termination measure for the head reduction of typable terms. Therefore, typable terms are head terminating – this is also called *correctness* of the type system (with respect to head reduction). Conversely, every head normal forms is typable. Additionally, one can show that the size of type derivations bounds the head size of head normal forms, and that there exists derivations having exactly the head size of the normal form, as in the example (1) above.

▶ **Proposition 5** (Typability of head normal forms). *Let h be a head normal form.*
1. Lax bounds for all pairs*: if $\Phi \rhd \Gamma \vdash h : \mathsf{L}$ then $|h|_{\mathtt{h}} \leq |\Phi|_{\mathtt{in}}$;*
2. Existence and exact bounds*: there exists a derivation $\Phi \rhd \Gamma \vdash h : \mathsf{L}$ such that $|h|_{\mathtt{h}} = |\Phi|_{\mathtt{in}}$.*

Typability of all head normal forms (Prop. 5.2) together with subject expansion (Prop. 3.2) implies the *completeness* of the type system: every head terminating term is typable.

▶ **Theorem 6** (Typability characterizes head normalization).
1. Correctness*: if $\Phi \rhd t$ then there exists a head normalizing evaluation $t \to_{\mathtt{h}}^{n} h$ with h normal and $2n + |h|_{\mathtt{h}} \leq |\Phi|_{\mathtt{in}}$.*
2. Completeness*: if $t \to_{\mathtt{h}}^{n} h$ is a head normalizing evaluation, then there exists a derivation $\Phi \rhd t$. In particular, there is a derivation Φ for which $2n + |h|_{\mathtt{h}} = |\Phi|_{\mathtt{in}}$.*

The quantitative bounds involve $2n$ rather than n because every β redex is typed in a type derivation Φ using two rules (λ and @). The type derivation captures only the head size of the normal form, because in general it ignores arguments, and so it cannot catch the inner size. For instance, for the head normal form $x\Omega$ of head size $|x\Omega|_{\mathtt{h}} = 1$ (but inner size $|x\Omega|_{\mathtt{in}} = 6$) the derivation (1) has inner size 1.

The head characterization theorem implies the following property of the semantics.

▶ **Theorem 7** (Adequacy of relational semantics for head reduction). *Let $\vec{x} = (x_1, \dots, x_n)$ be suitable for t. Then $[\![t]\!]_{\vec{x}}$ is non-empty if and only if t is $\to_{\mathtt{h}}$-normalizing.*

Summing up, multi types naturally model head reduction. De Carvalho's bounds from composable types, however, rest on normal forms, which are reached by leftmost reduction, rather than on head normal forms and head reduction. Therefore, the next section recalls how multi types relate to leftmost reduction and normal forms.

4 Bounds From Derivations Via (Unitary) Shrinking

In this section, we recall how to extend the results of the previous section to leftmost reduction $\to_{\mathtt{lo}}$ and full normal forms, via the so called *shrinking* constraint. We follow the presentation of Accattoli et al. [7] (removing some of the aspects of their work that are not relevant here), but the definition of shrinking judgements is standard and not due to [7], see for instance Krivine's book [44], de Carvalho [44, 27], Kesner and Ventura [40], or Bucciarelli et al. [20].

The Need for Shrinking. Consider the derivation of end sequent $x : [\mathbf{0} \multimap \mathsf{L}] \vdash x\Omega : \mathsf{L}$ in (1). Since $x\Omega$ is $\to_{\mathtt{lo}}$-diverging, this derivation has to be excluded somehow. The problem here is that since x has an erasing type – that is an arrow type with $\mathbf{0}$ on the left – then the diverging subterm Ω does not get typed. Excluding the use of $\mathbf{0}$ is too drastic, because the paradigmatic erasing term $\lambda y.x$ is normal and can be typed only with $x : [\mathsf{L}] \vdash \lambda y.x : \mathbf{0} \multimap \mathsf{L}$.

The idea is that only *some* occurrences of $\mathbf{0}$ are dangerous. The given examples seem to suggest that if $\mathbf{0}$ occurs on the right side of $\vdash$ it is fine, while if it occurs in the typing context it is not. Things are subtler. Extending example (1) with an abstraction, one obtains the $\to_{\mathtt{lo}}$-diverging term $\lambda x.x\Omega$ and the typing $\vdash \lambda x.x\Omega : [\mathbf{0} \multimap \mathsf{L}] \multimap \mathsf{L}$, that show that $\mathbf{0}$ can be

dangerous also on the right of $\vdash$. The dangerous occurrences of $\mathbf{0}$ turn out to be those on the left of an *even number of arrows*, considering the $\vdash$ symbols as an arrow. This is formalized by the *shrinking* constraint, which allows one to characterize leftmost termination.

Defining Shrinking. There are two mutually defined notions of shrinking types, *left* and *right*, the key point of which is that right multi types *cannot be empty* (note $n \geq 1$), so that $\mathbf{0}$ is forbidden on the left of top arrows $\multimap$ for left linear types. Their definition follows:

Left and right (shrinking) types

$$\begin{array}{rrcl} \text{Right linear type} & \mathsf{L}^{\mathrm{R}} & ::= & \mathsf{X} \in \mathtt{TyVars} \mid \mathsf{M}^{\mathrm{L}} \multimap \mathsf{L}^{\mathrm{R}} \\ \text{Left linear type} & \mathsf{L}^{\mathrm{L}} & ::= & \mathsf{X} \in \mathtt{TyVars} \mid \mathsf{M}^{\mathrm{R}} \multimap \mathsf{L}^{\mathrm{L}} \\ \text{Right multi type} & \mathsf{M}^{\mathrm{R}} & ::= & [\mathsf{L}^{\mathrm{R}}_1, \dots, \mathsf{L}^{\mathrm{R}}_n] \quad n \geq 1 \\ \text{Left multi type} & \mathsf{M}^{\mathrm{L}} & ::= & [\mathsf{L}^{\mathrm{L}}_1, \dots, \mathsf{L}^{\mathrm{L}}_n] \quad n \geq 0 \end{array}$$

The notions extend to type contexts and to derivations as follows:

- A type context $x_1 : \mathsf{M}_1, \dots, x_n : \mathsf{M}_n$ is *left* if each M_i is left;
- A derivation $\Phi \rhd \Gamma \vdash t : \mathsf{L}$ is *shrinking* if Γ is left and L is right.

For instance, $[\mathsf{X}]$ is both left and right, while $\mathbf{0}$ is left but not right, and $[\mathbf{0} \multimap \mathsf{X}]$ is right but not left. Note that the derivation in (1) is not shrinking. By adding the shrinking constraint, we can now characterize leftmost normalization with multi types, with quantitative bounds involving the inner size of the normal form.

▶ **Theorem 8** (Shrinking typability characterizes leftmost normalization, [7]).

1. Correctness*: if $\Phi \rhd t$ is a shrinking derivation, then there exists a normalizing evaluation $t \to_{\mathtt{lo}}^{n} f$ with f normal and $2n + |f|_{\mathtt{in}} \leq |\Phi|_{\mathtt{in}}$.*
2. Completeness*: if $t \to_{\mathtt{lo}}^{*} f$ is a normalizing evaluation, then there exists a shrinking derivation $\Phi \rhd t$.*

Unitary Shrinking. Shrinking is enough to ensure termination, but not to capture the exact number of steps to normal form together with the exact size of the normal form. The point is somewhat dual to shrinkingness, as it concerns arguments that have to be typed, but that should not be typed *too many times*. Consider the evaluation $y(\mathsf{I}z) \to_{\mathtt{lo}} yz$ that involves 1 leftmost step and produces a normal form of inner size 1. The following shrinking derivation types the argument $\mathsf{I}z$ twice (the easy derivation of $z : [\mathsf{X}] \vdash \mathsf{I}z : \mathsf{X}$ of inner size 2 is omitted), instead of once, and it has size 5, instead of the required 3 (obtained as 2*1+1):

$$\dfrac{\dfrac{}{y : [[\mathsf{X},\mathsf{X}] \multimap \mathsf{Y}] \vdash y : [\mathsf{X},\mathsf{X}] \multimap \mathsf{Y}}\,\mathsf{ax} \qquad \dfrac{[z : [\mathsf{X}] \vdash \mathsf{I}z : \mathsf{X}]_{i=1,2}}{z : [\mathsf{X},\mathsf{X}] \vdash \mathsf{I}z : [\mathsf{X},\mathsf{X}]}\,\mathsf{many}}{y : [[\mathsf{X},\mathsf{X}] \multimap \mathsf{Y}], z : [\mathsf{X},\mathsf{X}] \vdash y(\mathsf{I}z) : \mathsf{Y}}\,@ \tag{2}$$

To obtain exact bounds, one needs *unitary shrinking* types and derivations, that type arguments of normal forms only once, obtained by constraining some multi-sets – the right ones – to be singletons. The definition follows:

Unitary left and right (shrinking) types

$$\begin{array}{rrcl} \text{Unitary right linear types} & \mathsf{L}^{\mathrm{UR}} & ::= & \mathsf{X} \in \mathtt{TyVars} \mid \mathsf{M}^{\mathrm{UL}} \multimap \mathsf{L}^{\mathrm{UR}} \\ \text{Unitary left linear types} & \mathsf{L}^{\mathrm{UL}} & ::= & \mathsf{X} \in \mathtt{TyVars} \mid \mathsf{M}^{\mathrm{UR}} \multimap \mathsf{L}^{\mathrm{UL}} \\ \text{Unitary right multi types} & \mathsf{M}^{\mathrm{UR}} & ::= & [\mathsf{L}^{\mathrm{UR}}] \\ \text{Unitary left multi types} & \mathsf{M}^{\mathrm{UL}} & ::= & [\mathsf{L}^{\mathrm{UL}}_1, \dots, \mathsf{L}^{\mathrm{UL}}_n] \quad n \geq 0 \end{array}$$

The notions extend to type contexts and to derivations as expected:

- A type context $x_1 : \mathsf{M}_1, \dots, x_n : \mathsf{M}_n$ is *unitary left* if each M_i is unitary left;
- A derivation $\Phi \rhd \Gamma \vdash t : \mathsf{L}$ is *unitary shrinking* if Γ is unitary left and L is unitary right.

For instance, the derivation in (2) is not unitary shrinking, because the multi type $[[\mathsf{X}, \mathsf{X}] \multimap \mathsf{Y}]$ of y is not unitary left, since $[\mathsf{X}, \mathsf{X}]$ is not unitary right. A derivable unitary shrinking typing for $y(\mathsf{I}z)$ is $y : [[\mathsf{X}] \multimap \mathsf{Y}], z : [\mathsf{X}] \vdash y(\mathsf{I}z) : \mathsf{Y}$, obtained via a derivation of inner size 3.

The following refinement of the shrinking characterization theorem (Thm. 8) holds.

▶ **Theorem 9** (Unitary shrinking typability measures leftmost evaluation, [7]).

1. Correctness*: if $\Phi \rhd t$ is a unitary shrinking derivation, then there exists a normalizing evaluation $t \to_{\mathtt{lo}}^{n} f$ with f normal and $2n + |f|_{\mathtt{in}} = |\Phi|_{\mathtt{in}}$.*
2. Completeness*: if $t \to_{\mathtt{lo}}^{*} f$ is a normalizing evaluation, then there exists a unitary shrinking derivation $\Phi \rhd t$.*

Normal Forms. The proof of the last theorem rests on two properties of normal forms that it is useful to state explicitly, for comparison with the study of the next sections.

▶ **Proposition 10** (Unitary shrinking derivations and normal forms, [7]). *Let f be normal.*

1. Lax bounds*: if $\Phi \rhd f$ is a shrinking derivation then $|f|_{\mathtt{in}} \leq |\Phi|_{\mathtt{in}}$;*
2. Exact bounds*: there exists a unitary shrinking derivation $\Phi \rhd f$ such that $|f|_{\mathtt{in}} = |\Phi|_{\mathtt{in}}$.*

5 Bounds from Types

In this section, we recall the bounds on the size of normal forms that can be extracted from *types* rather than from *type derivations*.

Bounds from Types. The types appearing in the final judgement of a shrinking derivation for t bound the inner size $|f|_{\mathtt{in}}$ of the normal form f of t, according to a notion of *type size* given below, and independently of the derivation itself. For example, consider the easily derivable (unitary shrinking) derivation $\Phi \rhd\vdash \delta : [[\mathsf{X}] \multimap \mathsf{X}, \mathsf{X}] \multimap \mathsf{X}$ for $\delta = \lambda x.xx$. There are two arrows in the type (judgement) and the normal form has inner size two. Of course, one also has to take into account the arrow symbols appearing in the typing context, when present.

Note, however, that types – even unitary shrinking ones – in general do not provide exact bounds: taking the derivation of Φ for δ and substituting X with $[\mathsf{Y}] \multimap \mathsf{Y}$ everywhere in Φ one obtains a unitary shrinking derivation Ψ having the same size of Φ but final (still unitary shrinking) judgement:

$$\Psi \rhd \vdash \delta : [[[\mathsf{Y}] \multimap \mathsf{Y}] \multimap [\mathsf{Y}] \multimap \mathsf{Y}, [\mathsf{Y}] \multimap \mathsf{Y}] \multimap [\mathsf{Y}] \multimap \mathsf{Y}$$

which has six arrows while $|\delta|_{\mathtt{in}} = 2$.

▶ **Definition 11** (Type size). *The size $|\cdot|$ of types and typing contexts is defined as follows:*

$$\begin{array}{lllll} \textsc{Types} & |\mathsf{X}| := 0 & |\mathsf{M} \multimap \mathsf{L}| := |\mathsf{M}| + |\mathsf{L}| + 1 & |[\mathsf{L}_1, \dots, \mathsf{L}_n]| := \sum_{i=1}^{n} |\mathsf{L}_i| \\ \textsc{Type ctxs} & |\epsilon| := 0 & |x : \mathsf{M}; \Gamma| := |\mathsf{M}| + |\Gamma| & \end{array}$$

Clearly, $|\mathsf{T}| \geq 0$ and $|\mathsf{M}| = 0$ if and only if M is a possibly empty multi set of type variables.

Given a type context $\Gamma = x_1 : \mathsf{M}_1, \dots, x_n : \mathsf{M}_n$ we often consider the list of its types, noted $\hat{\Gamma} := (\mathsf{M}_1, \dots, \mathsf{M}_n)$. Since any list of multi types $(\mathsf{M}_1, \dots, \mathsf{M}_n)$ can be seen as extracted from a type context Γ, we use the notation $\hat{\Gamma}$ for lists of multi types. The *size* of a list of multi types is $|(\mathsf{M}_1, \dots, \mathsf{M}_n)| := \sum_{i=1}^{n} |\mathsf{M}_i|$, and that of the conclusion of a derivation $\pi \rhd \Gamma \vdash e : \mathsf{L}$ is $|(\hat{\Gamma}, \mathsf{L})| := |\hat{\Gamma}| + |\mathsf{L}|$. Clearly, $\mathsf{dom}(\Gamma) = \emptyset$ implies $|\hat{\Gamma}| = 0$.

▶ **Proposition 12** (Shrinking types bound the size of normal forms, [7]). *Let f be a normal form.*

1. Lax bounds for all types*: if $\Phi \rhd \Gamma \vdash f : \mathsf{L}$ is a shrinking derivation then $|f|_{\mathtt{in}} \leq |(\hat{\Gamma}, \mathsf{L})|$;*
2. Exact bounds for special types*: there exists a unitary shrinking derivation $\Phi \rhd \Gamma \vdash f : \mathsf{L}$ such that $|f|_{\mathtt{in}} = |(\hat{\Gamma}, \mathsf{L})|$.*

6 Dissecting Bounds From Types via Skeletons and Dry Judgements

In this section, we decompose and elaborate over the bounds on the size of normal forms extracted from types given in the previous section. The analysis is the main contribution of this paper. In particular, we develop notions and tools that shall be used in the next section to understand the issues concerning how to extract exact bounds from composable types.

Types Bound the Size of Derivations. The first observation is that the lax bounds of Prop. 12.1 are a consequence of the more general fact that types bound the size of derivations, proved next, together with the already proved fact that derivations bound the size of normal forms (Prop. 10). The second point of the following proposition is the main statement, the first one is an auxiliary one that is needed for the proof to go through.

▶ **Proposition 13** (Types bound the size of derivations for normal forms). *Let $\Phi \rhd \Gamma \vdash t : \mathsf{T}$ be a derivation.*

1. Neutral: *if t is a neutral term then $|\Phi|_{\mathtt{in}} \leq |\hat{\Gamma}| - |\mathsf{T}|$.*
2. Normal: *if t is a normal form then $|\Phi|_{\mathtt{in}} \leq |\hat{\Gamma}| + |\mathsf{T}|$.*

Proof. By mutual induction on the definition of neutral and normal terms, followed by an induction on the type derivation Φ.

1. *t is a neutral term.* Cases of the last rule:
   - *Rule* many. Then T is a multi type $\mathsf{M} = [\mathsf{L}_i]_{i\in I}$ and the last rule is necessarily many. So, necessarily, for some finite set of indices I,

     $$\Phi = \dfrac{[\,\Phi_i \rhd \Gamma_i \vdash t : \mathsf{L}_i]_{i\in I}}{\uplus_{i\in I}\Gamma_i \vdash t : [\mathsf{L}_i]_{i\in I}}\,\mathsf{many}$$

     where $\Gamma = \uplus_{i\in I}\Gamma_i$. By *i.h.* (on Φ_i), $|\Phi_i|_{\mathtt{in}} \leq |\hat{\Gamma_i}| - |\mathsf{L}_i|$, thus $|\Phi|_{\mathtt{in}} = \sum_{i\in I} |\Phi_i|_{\mathtt{in}} \leq \sum_{i\in I} |\hat{\Gamma_i}| - \sum_{i\in I} |\mathsf{L}_i| = |\hat{\uplus_{i\in I}\Gamma_i}| - |[\mathsf{L}_i]_{i\in I}| = |\hat{\Gamma}| - |\mathsf{M}|$.
   - *Rule* ax, that is, $t = x$. Then:

     $$\Phi = \dfrac{}{x : [\mathsf{L}] \vdash x : \mathsf{L}}\,\mathsf{ax}$$

     where $\mathsf{T} = \mathsf{L}$ and $\Gamma = x : [\mathsf{L}]$. Since $|\Phi|_{\mathtt{in}} = 0$ and $|\mathsf{T}| = |\mathsf{L}| = |[\mathsf{L}]| = |\hat{\Gamma}|$, then $|\Phi|_{\mathtt{in}} = 0 = |\hat{\Gamma}| - |\mathsf{T}|$.
   - *Rule* @, that is, $t = nf$. Then necessarily:

     $$\Phi = \dfrac{\Phi_n \rhd \Delta \vdash n : \mathsf{N} \multimap \mathsf{L} \qquad \Phi_f \rhd \Sigma \vdash f : \mathsf{N}}{\Delta \uplus \Sigma \vdash nf : \mathsf{L}}\,@$$

     where $\mathsf{T} = \mathsf{L}$ and $\Gamma = \Delta \uplus \Sigma$. By *i.h.* (on the definition of neutral terms and normal forms), $|\Phi_n|_{\mathtt{in}} \leq |\hat{\Delta}| - |\mathsf{N} \multimap \mathsf{L}| = |\hat{\Delta}| - |\mathsf{N}| - |\mathsf{L}| - 1$ and $|\Phi_f|_{\mathtt{in}} \leq |\hat{\Sigma}| + |\mathsf{N}|$. Therefore,

     $$\begin{array}{rcll} |\Phi|_{\mathtt{in}} & = & |\Phi_n|_{\mathtt{in}} + |\Phi_f|_{\mathtt{in}} + 1 & \leq_{i.h.} \; |\hat{\Delta}| - |\mathsf{N}| - |\mathsf{L}| - 1 + |\Phi_f|_{\mathtt{in}} + 1 \\ & = & |\hat{\Delta}| - |\mathsf{N}| - |\mathsf{L}| + |\Phi_f|_{\mathtt{in}} & \leq_{i.h.} \; |\hat{\Delta}| - |\mathsf{N}| - |\mathsf{L}| + |\hat{\Sigma}| + |\mathsf{N}| \\ & = & |\hat{\Delta}| + |\hat{\Sigma}| - |\mathsf{L}| & = \; |\hat{\Gamma}| - |\mathsf{T}|. \end{array}$$

2. *t is a normal form.* If t is a neutral term, then the statement follows from Point 1, which stronger than the statement that we need to prove. Otherwise, t is an abstraction. If the last rule is many then we reason exactly as for neutral terms. The remaining cases is when the last rule is λ, that is, $t = \lambda x.f$ and Φ is necessarily of the form:

$$\frac{\Phi' \rhd \Delta \vdash f : \mathsf{L}'}{\Delta \setminus\!\setminus x \vdash \lambda x.f : \Delta(x) \multimap \mathsf{L}'}\,\lambda$$

where $\mathsf{T} = \Delta(x) \multimap \mathsf{L}'$ and $\Gamma = \Delta \setminus\!\setminus x$. By *i.h.*,

$$|\Phi'|_{\mathtt{in}} \leq_{i.h.} \quad |\hat{\Delta}| + |\mathsf{L}'| \quad = \quad |\Delta \mathbin{\hat{\setminus\!\setminus}} x| + |\Delta(x)| + |\mathsf{L}'| \quad = \quad |\Delta \mathbin{\hat{\setminus\!\setminus}} x| + |\Delta(x) \multimap \mathsf{L}'| - 1$$

Therefore,

$$\begin{array}{rclcl} |\Phi|_{\mathtt{in}} & = & |\Phi'|_{\mathtt{in}} + 1 & \leq & |\Delta \mathbin{\hat{\setminus\!\setminus}} x| + |\Delta(x) \multimap \mathsf{L}'| - 1 + 1 \\ & = & |\Delta \mathbin{\hat{\setminus\!\setminus}} x| + |\Delta(x) \multimap \mathsf{L}'| & = & |\hat{\Gamma}| + |\mathsf{T}|. \end{array}$$

◀

Note that the bound holds for *every* derivation, without requiring them to be shrinking. This fact means that the connection between types and derivations is stronger than the one between derivations and normal forms. Note also that the bound does *not* hold for *head* normal forms, as can be seen by inspecting examples (1) (p. 7) and (2) (p. 9).

Exact Bounds from Types. We now turn our attention to exact bounds. Having showed that bounds for normal forms factor through bounds for derivations (Prop. 13), we actually turn to exact bounds for *derivations*, from types. The idea, as usual, is that exact bounds are given by types of minimal size. To describe such minimal types we shall use a modified *dry* typing system for normal forms related to principal judgements, that shall derive only minimal types. Additionally, we use a relation between derivations having the same structure but assigning possibly different types, also considered by de Carvalho.

Skeleton Equivalence. We formalize the notion of derivations having the same *skeleton*, that is, the same *mute* structure. Skeleton equivalence $\sim$ relates derivations having the same rules arranged in the same way, but not necessarily having the same types.

▶ **Definition 14** (Skeleton equivalence). *Let t be a term. Two derivations $\Phi \rhd t$ and $\Psi \rhd t$ are* skeleton equivalent*, noted $\Phi \sim \Psi$, if they end with the same kind of rule and the derivations on the premises are $\sim$-equivalent, namely they fall in one of the following cases:*

- *Both Φ and Ψ are axioms.*
- *Both Φ and Ψ end with rule @, their two left premises Φ_l and Ψ_l satisfy $\Phi_l \sim \Psi_l$, and their right premises Φ_r and Ψ_r satisfy $\Phi_r \sim \Psi_r$ – similarly for rules λ.*
- *Both Φ and Ψ end with a rule* many *with n premises and there is a permutation ρ of $\{1, \ldots, n\}$ such that the i-th premise Φ_i of Φ and the $\rho(i)$-th premise $\Psi_{\rho(i)}$ of Ψ satisfy $\Phi_i \sim \Psi_{\rho(i)}$ for $i \in \{1, \ldots, n\}$.*

The next lemma shows that skeleton equivalence preserves more or less everything one can imagine, but types. We denote by $\#m$ the cardinality of a multiset m.

▶ **Lemma 15** (Skeletal invariants). *Let $\Phi \rhd \Gamma \vdash t : \mathsf{T}$ and $\Psi \rhd \Delta \vdash t : \mathsf{T}'$ be two derivations such that $\Phi \sim \Psi$. Then $|\Phi|_{\mathtt{in}} = |\Psi|_{\mathtt{in}}$, $\mathsf{dom}(\Gamma) = \mathsf{dom}(\Delta)$, $\#(\Gamma(x)) = \#(\Delta(x))$ for every variable x, and T is a multi type if and only if T' is, and in that case $\#\mathsf{T} = \#\mathsf{T}'$. Moreover, Φ is shrinking (resp. unitary shrinking) if and only if Ψ is.*

Proof. Straightforward induction on Φ. ◀

$$\frac{}{x:[\mathsf{X}] \vdash^{\mathsf{d}} x:\mathsf{X}}\,\mathsf{ax}^*$$

$$\frac{\Phi \rhd \Gamma \vdash^{\mathsf{d}} n:\mathsf{X} \quad \Psi \rhd \Delta \vdash^{\mathsf{d}} f:\mathsf{M} \quad \mathsf{Y} \text{ fresh}, \Phi \# \Psi}{(\Gamma\{\mathsf{X}\leftarrow(\mathsf{M} \multimap \mathsf{Y})\} \uplus \Delta) \vdash^{\mathsf{d}} nf:\mathsf{Y}}\,@^*$$

$$\frac{\Gamma \vdash^{\mathsf{d}} f:\mathsf{L}}{\Gamma \setminus\!\setminus x \vdash^{\mathsf{d}} \lambda x.f:\Gamma(x) \multimap \mathsf{L}}\,\lambda^*$$

$$\frac{\left[\Phi_i \rhd \Gamma_i \vdash^{\mathsf{d}} f:\mathsf{L}_i\right]_{i\in I} \quad \#_{i\in I}\,\Phi_i}{\biguplus_{i\in I}\Gamma_i \vdash^{\mathsf{d}} f:[\mathsf{L}_i]_{i\in I}}\,\mathsf{many}^*$$

Figure 2 Dry multi type system for normal forms.

Principal and Dry Judgements. Simple types admits *principal judgements* (or typings), that is, for every term t there exists a principal judgement $\Gamma \vdash t:A$ such that for every other judgement $\Delta \vdash t:B$ for t there exists a type substitution σ such that $\Gamma\sigma = \Delta$ and $A\sigma = B$. Multi types do not have principal judgements, since there is no *single* judgement for a term that subsumes all others *up to substitutions*. The literature has studied a weakened notion of principal judgement, subsuming all judgements up to substitution *and* up to another (very technical) operation called *expansion*, which – roughly – duplicates multi sets [22, 50, 43].

What we are going to do next, intuitively, is following the other natural route when principal judgements do not exist: we study a notion of principal *set* of special judgements for a term t, called *dry* judgements, which are such that every ordinary judgement for t can be seen as a dry judgement up to substitution. In fact, we only study this property for normal forms, and we also relate the derivations producing those judgements. We need some definitions.

Supports and Substitutions. The *support* of a type derivation $\Phi \rhd \Gamma \vdash t:\mathsf{T}$ is the set $\mathtt{TyVars}(\Phi) := \{\mathsf{X} \mid \mathsf{X} \text{ occurs in } \Phi\}$ of type variables appearing in Φ, and the *final support* is the set $\mathtt{TyVarsF}(\Phi) := \{\mathsf{X} \mid \mathsf{X} \text{ occurs in } \Gamma \text{ or } \mathsf{T}\}$ of type variables appearing in the last judgement of Φ. We write $\Phi \# \Psi$ as a shortcut for $\mathtt{TyVars}(\Phi) \cap \mathtt{TyVars}(\Psi) = \emptyset$ and given $\{\Phi_i\}_{i\in I}$ we write $\#_{i\in I}\,\Phi_i$ when $\mathtt{TyVars}(\Phi_h) \cap \mathtt{TyVars}(\Phi_k) = \emptyset$ for any two distinct $h, k \in I$.

A type substitution σ is a function from type variables to linear types that is the identity but a for finite number of type variables. It is extended to act on types, multi types, type contexts, and derivations as expected.

Dry Judgements. Dry judgements for normal forms are derived using the rules in Fig. 2. There are three key points. Firstly, only normal forms are typable. Secondly, neutral terms are always typed with a type variable (which is minimal) and when they are applied (in rule $@^*$) their type is *enlarged* on-the-fly via a type substitution $\{\mathsf{X}\leftarrow(\mathsf{M} \multimap \mathsf{Y})\}$ depending on the type of the argument. Thirdly, the system uses many type variables, and for the rules with more than one premise (*i.e.* $@^*$ and many^*) it requires them to have disjoint supports. This is where having countably many type variables plays a role, as having only a finite number would not allow one to prove the *subsumption up to substitutions* property of dry derivations, given by Thm. 19.2 below.

Dry derivations can be seen as standard derivations, as the second point of the next lemma states. It is obtained using the straightforward fact that standard derivations are stable by type substitutions. Note the use of skeleton equivalence.

▶ **Lemma 16.** *Let t be a term and f be a normal form.*

1. Substitutivity for standard*: if $\Phi \rhd \Gamma \vdash t:\mathsf{L}$ then for any linear type L' there exists $\Phi_{\{\mathsf{X}\leftarrow\mathsf{L}'\}} \rhd \Gamma\{\mathsf{X}\leftarrow\mathsf{L}'\} \vdash t:\mathsf{L}\{\mathsf{X}\leftarrow\mathsf{L}'\}$ such that $\Phi \sim \Phi_{\{\mathsf{X}\leftarrow\mathsf{L}'\}}$.*
2. Dry derivations are standard*: if $\Phi \rhd \Gamma \vdash^{\mathsf{d}} f:\mathsf{L}$ then there exists $\Psi \rhd \Gamma \vdash f:\mathsf{L}$ such that $\Phi \sim \Psi$.*

Proof. The first point is a straightforward induction on Φ. The second point is by induction on Φ. The only rule of the dry system that is not a rule of the standard system is $@^*$:

$$\frac{\Phi_n \rhd \Gamma_n \vdash^{\mathsf{d}} n : \mathsf{X} \quad \Phi_{f'} \rhd \Gamma_{f'} \vdash^{\mathsf{d}} f' : \mathsf{M} \quad \mathsf{Y} \text{ fresh}, \Phi \# \Psi}{\Gamma_n\{\mathsf{X} \leftarrow (\mathsf{M} \multimap \mathsf{Y})\} \uplus \Gamma_{f'} \vdash^{\mathsf{d}} nf' : \mathsf{Y}} @^*$$

with $f = nf'$, $\Gamma = \Gamma_n\{\mathsf{X} \leftarrow (\mathsf{M} \multimap \mathsf{Y})\} \uplus \Gamma_{f'}$, $\mathsf{T} = \mathsf{Y}$. By *i.h.*, there exist $\Psi_n \rhd \Gamma_n \vdash n : \mathsf{X}$ and $\Psi_{f'} \rhd \Gamma_{f'} \vdash f' : \mathsf{M}$. By Point 1, there exists a derivation: $\Psi_n\{\mathsf{X} \leftarrow (\mathsf{M} \multimap \mathsf{Y})\} \rhd \Gamma_n\{\mathsf{X} \leftarrow (\mathsf{M} \multimap \mathsf{Y})\} \vdash n : \mathsf{M} \multimap \mathsf{Y}$ Then we build Ψ as follows:

$$\frac{\Psi_n\{\mathsf{X} \leftarrow (\mathsf{M} \multimap \mathsf{Y})\} \rhd \Gamma_n\{\mathsf{X} \leftarrow (\mathsf{M} \multimap \mathsf{Y})\} \vdash n : \mathsf{M} \multimap \mathsf{Y} \quad \Psi_{f'} \rhd \Gamma_{f'} \vdash f' : \mathsf{M}}{\Gamma_n\{\mathsf{X} \leftarrow (\mathsf{M} \multimap \mathsf{Y})\} \uplus \Gamma_{f'} \vdash nf' : \mathsf{Y}} @$$

Skeleton equivalence of Φ and Ψ follows immediately from the skeleton equivalences of the *i.h.* plus the one of Point 1. ◀

Next, we prove that types in dry judgements are always minimal and – crucially – capture the size of the derivation itself. This is obtained via a strong property for type variables in dry judgements, reminiscent of similar properties in multiplicative linear logic, and enforced by the requirements about disjoint supports in the derivation rules.

▶ **Proposition 17.** *Let f be a normal form.*

1. Dry derivations and variable types occurrences*: if $\Phi \rhd \Gamma \vdash^{\mathsf{d}} f : \mathsf{T}$ then X has exactly two occurrences in (Γ, T) for every $\mathsf{X} \in \mathtt{TyVarsF}(\Phi)$.*
2. Dry derivations are minimal*: if $\Phi \rhd \Gamma \vdash^{\mathsf{d}} f : \mathsf{L}$ then $|\Phi|_{\mathtt{in}} = |(\hat{\Gamma}, \mathsf{L})|$.*

Proof.

1. By induction on Φ, looking at its last rule. For ax^*, the statement evidently holds, and for λ^* and rule many^* it follows from the *i.h.*, since these rules preserve and do not introduce occurrences of type variable. If the last rule of Φ is $@^*$, then Φ has shape:

$$\Phi = \frac{\Phi_1 \rhd \Delta \vdash^{\mathsf{d}} n : \mathsf{X} \quad \Phi_2 \rhd \Sigma \vdash^{\mathsf{d}} f' : \mathsf{M} \quad \mathsf{Y} \text{ fresh}, \Phi \# \Psi}{\Delta\{\mathsf{X} \leftarrow (\mathsf{M} \multimap \mathsf{Y})\} \uplus \Sigma \vdash^{\mathsf{d}} nf' : \mathsf{Y}} @^*$$

   with $f = nf'$, $\mathsf{T} = \mathsf{Y}$, and $\Gamma = \Delta\{\mathsf{X} \leftarrow (\mathsf{M} \multimap \mathsf{Y})\} \uplus \Sigma$. By *i.h.*, X occurs exactly once in Δ, thus Y occurs exactly twice in (Γ, T). Note that $\mathtt{TyVarsF}(\Phi) = (\mathtt{TyVarsF}(\Phi_1) \setminus \{\mathsf{X}\}) \cup \mathtt{TyVarsF}(\Phi_2) \cup \{\mathsf{Y}\}$. By *i.h.*, each type variable in $\mathtt{TyVarsF}(\Phi_1) \setminus \{\mathsf{X}\}$ (resp. $\mathtt{TyVarsF}(\Phi_2)$) occurs exactly twice in Δ (resp. (Σ, M)). Moreover, by hypothesis $\mathtt{TyVars}(\Phi_1) \cap \mathtt{TyVars}(\Phi_2) = \emptyset$, so each such type variable occurs exactly twice in (Γ, T).
2. By induction on Φ, looking at its last rule. Cases:
   - *Rule* ax^*: the statement holds because $|\Phi|_{\mathtt{in}} = 0 = |([\mathsf{X}], \mathsf{X})|$.
   - *Rule* many^*: it follows from the *i.h.*
   - *Rule* λ^*: it follows from the *i.h.* because rule λ^* add 1 to the size of the derivation, but the the size of the judgement also grows of 1, because of the introduced arrow.
   - *Rule* $@^*$: then Φ has the following shape:

$$\frac{\Phi_1 \rhd \Delta \vdash^{\mathsf{d}} n : \mathsf{X} \quad \Phi_2 \rhd \Sigma \vdash^{\mathsf{d}} f : \mathsf{M} \quad \mathsf{Y} \text{ fresh}, \Phi \# \Psi}{\Delta\{\mathsf{X} \leftarrow (\mathsf{M} \multimap \mathsf{Y})\} \uplus \Sigma \vdash^{\mathsf{d}} nf : \mathsf{Y}} @^*$$

     with $\Gamma = \Delta\{\mathsf{X} \leftarrow (\mathsf{M} \multimap \mathsf{Y})\} \uplus \Sigma$. By Point 1, X occurs exactly once in Δ. Then we have:

$$\begin{array}{rcll} |\Phi|_{\mathtt{in}} & = & |\Phi_1|_{\mathtt{in}} + |\Phi_2|_{\mathtt{in}} + 1 & =_{i.h.} \quad |\Delta| + |\Sigma| + |\mathsf{M}| + 1 \\ & = & |\Delta| + |\Sigma| + |\mathsf{M} \multimap \mathsf{Y}| & =_{P.1} \quad |\Delta\{\mathsf{X} \leftarrow (\mathsf{M} \multimap \mathsf{Y})\}| + |\Sigma| \\ & = & |\Delta\{\mathsf{X} \leftarrow (\mathsf{M} \multimap \mathsf{Y})\}| + |\Sigma| + |\mathsf{Y}| & \end{array}$$

◀

Dry Representation. We now prove the key property of the analysis, which also justifies seeing dry derivations as defining a set of principal judgements. The idea is that every (standard) derivation Φ admits a dry derivation Ψ that is skeleton equivalent to Φ – which by the skeletal invariants above entails that they have the same size – and such that there is a substitution turning Ψ into Φ. We need an auxiliary lemma that shall also be useful in the next section, about renamings of dry derivations.

▶ **Lemma 18** (Dry derivations are stable by renaming). *Let $\Phi \rhd \Gamma \vdash^{\mathsf{d}} f : \mathsf{T}$ be a dry type derivation $\mathtt{TyVars}(\Phi) = \{\mathsf{X}_1, \dots, \mathsf{X}_n\}$ and $\mathsf{Y}_1, \dots, \mathsf{Y}_n$ be distinct type variables Φ. Then the derivation $\Phi_{\{\mathsf{X}_1,\dots,\mathsf{X}_n \leftarrow \mathsf{Y}_1,\dots,\mathsf{Y}_n\}}$ obtained by simultaneously replacing X_i with Y_i in Φ for $i \in \{1, \dots, n\}$ is a dry type derivation such that $\Phi \sim \Phi_{\{\mathsf{X}_1,\dots,\mathsf{X}_n \leftarrow \mathsf{Y}_1,\dots,\mathsf{Y}_n\}}$.*

Proof. Straightforward induction on Φ. ◀

▶ **Theorem 19.** *Let f be a normal form and $\Phi \rhd \Gamma \vdash f : \mathsf{T}$ be a type derivation.*

1. Dry representation*: there exists a dry derivation $\Psi \rhd \Delta \vdash^{\mathsf{d}} f : \mathsf{T}'$ such that $\Phi \sim \Psi$;*
2. Type substitution*: there exists a type substitution σ such that $\Delta\sigma = \Gamma$ and $\mathsf{T}'\sigma = \mathsf{T}$.*

Proof. By induction on Φ. Cases of the last rule:

- *Rule* ax. Then Φ is a ax rule of conclusion $x : [\mathsf{L}] \vdash x : \mathsf{L}$. The dry representation Ψ of Φ is a ax rule of conclusion $x : [\mathsf{X}] \vdash^{\mathsf{d}} x : \mathsf{X}$. The type substitution of the statement is $\{\mathsf{X} \leftarrow \mathsf{L}\}$.
- *Rule* λ: it follows by the *i.h.*
- *Rule* many: it follows by the *i.h.* Note that, by stability of dry derivations under renaming (Lemma 18), we can assume that all the derivations Ψ_i given by the *i.h.* are on disjoint supports, so that the constraint $\#_{i \in I}\, \Psi_i$ for rule many^* is satisfied.
- *Rule* @: then $f = nf'$ and Φ has the following shape:

$$\frac{\Phi_n \rhd \Gamma_n \vdash n : \mathsf{M} \multimap \mathsf{L} \quad \Phi_{f'} \rhd \Gamma_{f'} \vdash f' : \mathsf{M}}{\Gamma_n \uplus \Gamma_{f'} \vdash nf' : \mathsf{L}}\,@$$

with $\Gamma = \Gamma_n \uplus \Gamma_{f'}$ and $\mathsf{T} = \mathsf{L}$. About the dry representation, by *i.h.* there are dry derivations $\Psi_n \rhd \Delta_n \vdash^{\mathsf{d}} n : \mathsf{X}$ and $\Psi_{f'} \rhd \Delta_{f'} \vdash^{\mathsf{d}} f' : \mathsf{N}$ such that $\Phi_n \sim \Psi_n$ and $\Phi_{f'} \sim \Psi_{f'}$. By stability of dry derivations under renaming (Lemma 18), we can assume that $\Psi_n \# \Psi_n$. Then Ψ is obtained as follows:

$$\frac{\Psi_n \rhd \Delta_n \vdash^{\mathsf{d}} n : \mathsf{X} \quad \Psi_{f'} \rhd \Delta_{f'} \vdash^{\mathsf{d}} f' : \mathsf{N} \quad \mathsf{Y} \text{ fresh}, \Psi_n \# \Psi_n}{\Delta_n\{\mathsf{X} \leftarrow (\mathsf{N} \multimap \mathsf{Y})\} \uplus \Delta_{f'} \vdash^{\mathsf{d}} nf' : \mathsf{Y}}\,@^*$$

About the type substitution, by *i.h.*, there are substitutions σ_n and $\sigma_{f'}$ such that $\Delta_n\sigma_n = \Gamma_n$ and $\mathsf{X}\sigma_n = \mathsf{M} \multimap \mathsf{L}$, and $\Delta_{f'}\sigma_{f'} = \Gamma_{f'}$ and $\mathsf{N}\sigma_{f'} = \mathsf{M}$. We can assume that $\mathsf{dom}(\sigma_n) = \mathsf{dom}(\Delta_n)$ and $\mathsf{dom}(\sigma_{f'}) = \mathsf{dom}(\Delta_{f'})$, and we know that $\mathsf{dom}(\sigma_n) \cap \mathsf{dom}(\sigma_{f'}) = \emptyset$. Define the substitution σ as σ_n on $\mathsf{dom}(\sigma_n) \setminus \{\mathsf{X}\}$, as $\sigma_{f'}$ on $\mathsf{dom}(\sigma_{f'})$, and as $\{\mathsf{Y} \leftarrow \mathsf{L}\}$ on Y. Note that $\sigma_n(\mathsf{X}) = \mathsf{M} \multimap \mathsf{L}$ and let σ'_n be σ_n without $\{\mathsf{X} \leftarrow (\mathsf{M} \multimap \mathsf{L})\}$. Then:

$$\begin{array}{lcl} (\Delta_n\{\mathsf{X} \leftarrow (\mathsf{N} \multimap \mathsf{Y})\} \uplus \Delta_{f'})\sigma & = & \Delta_n\sigma'_n\{\mathsf{X} \leftarrow (\mathsf{N}\sigma_{f'} \multimap \mathsf{Y}\{\mathsf{Y} \leftarrow \mathsf{L}\})\} \uplus \Delta_{f'}\sigma_{f'} \\ & =_{i.h.} & \Delta_n\sigma'_n\{\mathsf{X} \leftarrow (\mathsf{M} \multimap \mathsf{L})\} \uplus \Gamma_{f'} \\ & = & \Delta_n\sigma_n \uplus \Gamma_{f'} \quad =_{i.h.} \quad \Gamma_n \uplus \Gamma_{f'} \quad = \quad \Gamma \end{array}$$

and $\mathsf{Y}\sigma = \mathsf{Y}\{\mathsf{Y} \leftarrow \mathsf{L}\} = \mathsf{L} = \mathsf{T}$. ◀

Removing Substitutions. In the previous theorem, the substitution part rests on the properties of dry derivations enabled by the countable number of type variables in the type system. We now show that a slightly weaker result is possible even with only one type variable and without dry derivations. The type substitution part shall not be recoverable, but the representation and the quantitative bounds are.

Let $\Gamma \vdash^1 t : \mathsf{L}$ denote a (standard) type derivation built using only 1-*types*, that is, types built using a single fixed type variable X.

▶ **Theorem 20** (Size representation). *Let f be a normal term and $\Phi \rhd \Gamma \vdash f : \mathsf{L}$ be a derivation. Then there exists a derivation $\Psi \rhd \Delta \vdash^1 f : \mathsf{L}'$ such that $\Psi \sim \Phi$ and $|\Psi|_{\mathtt{in}} = |\hat{\Delta}| + |\mathsf{L}'|$.*

The proof of the theorem in fact requires a stronger statement for the induction to go through, in particular having a separate point about neutral terms, for which a stronger property holds. See the technical report [1].

7 Bounds From Composable Types

In this section, we finally study bounds from composable types for leftmost evaluation and normal forms, which is the technical and neglected part of de Carvalho's work. To ease the study, we restrict to closed terms, so that type contexts disappear – de Carvalho does the same. There are however no issues in dealing with open terms.

Composable Types. De Carvalho's idea is that, given two normal forms t and u, one can extract bounds for tu by looking only at the types of t and u – that is, at $[\![t]\!]$ and $[\![u]\!]$ – because a derivation for tu is just the application of a derivation for t and one for u. We need to give a formal status to composable types, and we also need a notion of types that compose up to a type substitution.

▶ **Definition 21** (Composable pairs). *Let t and u be closed terms.*

- *A* (shrinking) composable pair *(of types) for t and u is a pair $p = (\mathsf{L}, \mathsf{M})$ such that $\mathsf{L} = \mathsf{M} \multimap \mathsf{L}' \in [\![t]\!]$, $\mathsf{M} \in [\![u]\!]$, and L' is right. The set of composable pairs of t and u is noted* $\mathtt{ShComPairs}(t, u)$.
- *A* (shrinking) composable pair up to substitution *for t and u is a pair $p = (\mathsf{L}, \mathsf{M})$ such that there exists a type substitution σ such that $(\mathsf{L}\sigma, \mathsf{M}\sigma) \in \mathtt{ShComPairs}(t, u)$. The set of composable pairs up to substitution of t and u is noted as* $\mathtt{ShComPairsSub}(t, u)$.

Note that if $(\mathsf{M} \multimap \mathsf{L}', \mathsf{M}) \in \mathtt{ShComPairs}(t, u)$ only L' is required to be a right shrinking type (as it is the only type in the judgement for tu after composition), while $\mathsf{M} \multimap \mathsf{L}'$ might very well not be a right shrinking type (if M is not left). The constraint that L' is right in the definition of $\mathtt{ShComPairs}(t, u)$ ensures the following property.

▶ **Lemma 22** (Normalization and composable types). *Let t and u be closed terms. Then tu $\to_{\mathtt{lo}}$-normalizes if and only if $\mathtt{ShComPairs}(t, u) \neq \emptyset$.*

Proof. If tu normalizes then by shrinking completeness (Thm. 8) there exists a shrinking derivation $\Phi \vdash tu : \mathsf{L}$ with L right. The last rule of Φ is @ and its premises give a composable pair $(\mathsf{M} \multimap \mathsf{L}, \mathsf{M})$ for t and u, for some M. Therefore, $\mathtt{ShComPairs}(t, u) \supset \{(\mathsf{M} \multimap \mathsf{L}, \mathsf{M})\} \neq \emptyset$.

Vice versa, if $\mathtt{ShComPairs}(t, u) \neq \emptyset$ then every composable pair induces a shrinking derivation for tu, by connecting the two derivations producing the composable pair via rule @. Thus, tu is typable. By shrinking correctness (Thm. 8), tu is $\to_{\mathtt{lo}}$-normalizing. ◀

Normal Forms and Composable Types. To warm up, we first show how to bound the size of normal forms from composable pairs.

▶ **Theorem 23** (Normal form bounds from composable types). *Let t and u be closed terms such that there is a normalizing evaluation $d : tu \to_{\mathtt{lo}}^{*} f$.*

1. Lax bounds*: $|f|_{\mathtt{in}} \leq |\mathsf{L}|$ for every composable pair $(\mathsf{M} \multimap \mathsf{L}, \mathsf{M}) \in \mathtt{ShComPairs}(t, u)$.*
2. Exact bounds from special pairs*: moreover, there exists a composable pair $(\mathsf{M} \multimap \mathsf{L}, \mathsf{M}) \in \mathtt{ShComPairs}(t, u)$ such that $|f|_{\mathtt{in}} = |\mathsf{L}|$.*

Proof.

1. Let $(\mathsf{M} \multimap \mathsf{L}, \mathsf{M}) \in \mathtt{ShComPairs}(t, u)$ be a composable pair. Then, there are derivations $\Phi_t \rhd\vdash t : \mathsf{M} \multimap \mathsf{L}$ and $\Phi_u \rhd\vdash u : \mathsf{M}$. We compose them as a derivation Φ for tu via rule @:

$$\Phi := \dfrac{\Phi_t \rhd\vdash t : \mathsf{M} \multimap \mathsf{L} \qquad \Phi_u \rhd\vdash u : \mathsf{M}}{\vdash tu : \mathsf{L}} @$$

   Note that the definition of composable pair guarantees that L is right (shrinking), so that Φ is shrinking. By shrinking correctness (Thm. 8), there is a derivation $\vdash f : \mathsf{L}$. Since shrinking types bound the size of normal forms (Prop. 12), $|f|_{\mathtt{in}} \leq |\mathsf{L}|$.
2. By Prop. 12.2, there exists a unitary shrinking derivation $\Psi \rhd\vdash f : \mathsf{L}$ for f such that $|\mathsf{L}| = |f|_{\mathtt{in}}$. Pulling back the final judgement of Ψ using subject expansion (Prop. 3.2), we obtain a derivation $\Theta \vdash tu : \mathsf{L}$. The last rule of Θ is @ and its premises give a composable pair $(\mathsf{M} \multimap \mathsf{L}, \mathsf{M})$ for t and u, for some M. ◀

Lax Evaluation Bounds from Composable Types. Now, we study how to additionally extract (bounds on) the number of leftmost step from a composable pair. Obtaining lax bounds is easy. The idea is that the composed type bounds the size of a derivation for tu, which in turns provides bounds about tu, as shown in Sect. 4. We also show that even composable pairs up to substitution yield bounds.

▶ **Theorem 24** (Lax bounds from composable types). *Let f and f' be closed normal terms such that $d : ff' \to_{\mathtt{lo}}^{*} f''$ with f'' normal. Then:*

1. Lax bounds and types*: $2|d| + |f''|_{\mathtt{in}} \leq |\mathsf{L}| + |\mathsf{M}| + 1$ for every composable pair $(\mathsf{L}, \mathsf{M}) \in \mathtt{ShComPairs}(f, f')$.*
2. Lax bounds and types, up to substitutions*: $2|d| + |f''|_{\mathtt{in}} \leq |\mathsf{L}| + |\mathsf{M}| + 1$ for every composable pair up to substitution $(\mathsf{L}, \mathsf{M}) \in \mathtt{ShComPairsSub}(f, f')$.*

Proof.

1. Let $(\mathsf{L}, \mathsf{M}) \in \mathtt{ShComPairs}(f, f')$ be a composable pair, which implies $\mathsf{L} = \mathsf{M} \multimap \mathsf{L}'$ for some right L'. Then, there are two derivations $\Phi_f \rhd\vdash f : \mathsf{M} \multimap \mathsf{L}'$ and $\Phi_{f'} \rhd\vdash f' : \mathsf{M}$. We compose them via a @ rule into a derivation Φ for ff':

$$\Phi := \dfrac{\Phi_f \rhd\vdash f : \mathsf{M} \multimap \mathsf{L}' \qquad \Phi_{f'} \rhd\vdash f' : \mathsf{M}}{\vdash ff' : \mathsf{L}'} @$$

   By definition of composable pair, L' is right shrinking, so that Φ is shrinking. By shrinking correctness (Thm. 8), $2|d| + |f''|_{\mathtt{in}} \leq |\Phi|_{\mathtt{in}} = |\Phi_f|_{\mathtt{in}} + |\Phi_{f'}|_{\mathtt{in}} + 1$. Now, since types bound the size of the derivation for normal terms (Prop. 13), we obtain $|\Phi_f|_{\mathtt{in}} \leq |\mathsf{M} \multimap \mathsf{L}'|$ and $|\Phi_{f'}|_{\mathtt{in}} \leq |\mathsf{M}|$. Therefore, $2|d| + |f''|_{\mathtt{in}} \leq |\Phi|_{\mathtt{in}} \leq |\mathsf{M} \multimap \mathsf{L}'| + |\mathsf{M}| + 1$, as required.
2. Let $(\mathsf{L}, \mathsf{M}) \in \mathtt{ShComPairsSub}(f, f')$ be a composable pair up to substitution, which implies that there exist a type substitution σ such that $\mathsf{L}\sigma = \mathsf{N} \multimap \mathsf{L}'$, $\mathsf{M}\sigma = \mathsf{N}$ for some right L' and some N. Then, there are two derivations $\Phi_f \rhd\vdash f : \mathsf{L}$ and $\Phi_{f'} \rhd\vdash f' : \mathsf{M}$. By

$$\dfrac{\dfrac{\dfrac{}{x:[[\mathsf{W}^2]\multimap\mathsf{W}^2]\vdash x:[\mathsf{W}^2]\multimap\mathsf{W}^2}\,\mathsf{ax}\quad\dfrac{\dfrac{}{x:[\mathsf{W}^2]\vdash x:\mathsf{W}^2}\,\mathsf{ax}}{x:[\mathsf{W}^2]\vdash x:[\mathsf{W}^2]}\,\mathsf{many}}{x:[[\mathsf{W}^2]\multimap\mathsf{W}^2,\mathsf{W}^2]\vdash xx:\mathsf{W}^2}\,@}{\Psi_\delta\rhd\vdash\lambda x.xx:[[\mathsf{W}^2]\multimap\mathsf{W}^2,\mathsf{W}^2]\multimap\mathsf{W}^2}\,\lambda\qquad\dfrac{\dfrac{\dfrac{}{y:[\mathsf{W}^2]\vdash y:\mathsf{W}^2}\,\mathsf{ax}}{\vdash\lambda y.y:[\mathsf{W}^2]\multimap\mathsf{W}^2}\,\lambda\quad\dfrac{\dfrac{}{y:[\mathsf{W}]\vdash y:\mathsf{W}}\,\mathsf{ax}}{\vdash\lambda y.y:[\mathsf{W}]\multimap\mathsf{W}}\,\lambda}{\Psi_{\mathsf{I}}\rhd\vdash\lambda y.y:[[\mathsf{W}^2]\multimap\mathsf{W}^2,\mathsf{W}^2]}\,\mathsf{many}}{\vdash\delta\mathsf{I}:\mathsf{W}^2}\,@$$

Figure 3 Unitary shrinking derivation $\Psi_{\delta\mathsf{I}}$ of minimal type for $\delta\mathsf{I}$, where $\mathsf{W}^2 \coloneqq [\mathsf{W}]\multimap\mathsf{W}$.

Lemma 16.1, applying σ to Φ_f and $\Phi_{f'}$ we obtain two derivations $\Psi_f \rhd\vdash f:\mathsf{N}\multimap\mathsf{L}'$ and $\Psi_{f'} \rhd\vdash f':\mathsf{N}$ such that $\Phi_f \sim \Psi_f$ and $\Phi_{f'} \sim \Psi_{f'}$. We compose them via a @ rule into a derivation Ψ for ff':

$$\Psi \coloneqq \dfrac{\Psi_f \rhd\vdash f:\mathsf{N}\multimap\mathsf{L}' \qquad \Psi_{f'} \rhd\vdash f':\mathsf{N}}{\vdash ff':\mathsf{L}'}\,@$$

Since L' is right, Ψ is shrinking. By shrinking correctness (Thm. 8), $2|d|+|f''|_{\mathtt{in}} \leq |\Psi|_{\mathtt{in}} = |\Psi_f|_{\mathtt{in}}+|\Psi_{f'}|_{\mathtt{in}}+1$. By the skeletal invariants (Lemma 15), we obtain $|\Psi_f|_{\mathtt{in}}+|\Psi_{f'}|_{\mathtt{in}}+1 = |\Phi_f|_{\mathtt{in}}+|\Phi_{f'}|_{\mathtt{in}}+1$. Since types bound the size of the derivation for normal terms (Prop. 13), $|\Phi_f|_{\mathtt{in}} \leq |\mathsf{L}|$ and $|\Phi_{f'}|_{\mathtt{in}} \leq |\mathsf{M}|$. Therefore, $2|d| + |f''|_{\mathtt{in}} \leq |\mathsf{L}| + |\mathsf{M}| + 1$, as required. ◀

Note that the hypotheses for bounding evaluation lengths are stronger than for bounding normal forms (Thm. 23), as the two applied terms f and f' are required to be normal. If the two composed terms are not normal, then there is no way to measure the extra steps to their normal forms using their types, because types are invariant by reduction. The stronger hypotheses limit the scope of the result: if $t \coloneqq \lambda x.x\Omega f$ with f normal and $u \coloneqq \lambda y.\lambda z.z$ then $tu \to^3_{\mathtt{lo}} f$ and Thm. 23 can be applied, while Thm. 24 cannot, because of the diverging Ω sub-term in t.

Exact Bounds from Composable Types. Now, the tricky point is how to obtain *exact* bounds. The problem is that for the application of two normal terms f and f', the minimal composable pair in general does *not* provide exact bounds, and – dually – minimal types for f and f' do *not* compose. The following example pinpoints the subtleties.

Key Example. Consider the unitary shrinking derivations Φ_δ and Φ_{I} of minimal types for $\delta \coloneqq \lambda x.xx$ and for $\mathsf{I} \coloneqq \lambda y.y$:

$$\Phi_\delta = \dfrac{\dfrac{\dfrac{}{x:[[\mathsf{Z}]\multimap\mathsf{Z}]\vdash x:[\mathsf{Z}]\multimap\mathsf{Z}}\,\mathsf{ax}\quad\dfrac{}{x:[\mathsf{Z}]\vdash x:\mathsf{Z}}\,\mathsf{ax}}{x:[[\mathsf{Z}]\multimap\mathsf{Z},\mathsf{Z}]\vdash xx:\mathsf{Z}}\,@}{\vdash\lambda x.xx:[[\mathsf{Z}]\multimap\mathsf{Z},\mathsf{Z}]\multimap\mathsf{Z}}\,\lambda \qquad \Phi_{\mathsf{I}} = \dfrac{\dfrac{}{y:[\mathsf{W}]\vdash y:\mathsf{W}}\,\mathsf{ax}}{\vdash\lambda y.y:[\mathsf{W}]\multimap\mathsf{W}}\,\lambda$$

Note that, pleasantly, $|\Phi_\delta|_{\mathtt{in}} = 2 = |[[\mathsf{Z}]\multimap\mathsf{Z},\mathsf{Z}]\multimap\mathsf{Z}| = |\delta|_{\mathtt{in}}$, and $|\Phi_{\mathsf{I}}|_{\mathtt{in}} = 1 = |[\mathsf{W}]\multimap\mathsf{W}| = |\mathsf{I}|_{\mathtt{in}}$. Now, consider the application $\delta\mathsf{I}$. Note that, unfortunately, the two obtained minimal types do *not* compose, and not just because they use different type variables: identifying Z and W would not be enough, one actually needs to identify Z with $\mathsf{W}^2 \coloneqq [\mathsf{W}]\multimap\mathsf{W}$. The unitary shrinking derivation $\Psi_{\delta\mathsf{I}}$ for $\delta\mathsf{I}$ with minimal types (which provides exact information for $\delta\mathsf{I}$) obtained in this way is in Fig. 3. Note that its sub-derivations Ψ_δ and Ψ_{I} for δ and I do *not* derive minimal types. The derivation $\Psi_{\delta\mathsf{I}}$ indeed is obtained by composing:

1. The variant Ψ_δ of Φ_δ which has the same exact structure of Φ_δ and where every occurrence of Z has been replaced by W^2, obtaining the type $[[[\mathsf{W}^2]\multimap\mathsf{W}^2,\mathsf{W}^2]]\multimap\mathsf{W}^2$,

2. With Ψ_{I}, which puts together two derivations for I, one being Φ_{I} (of type W^2), and one being the variant Φ'_{I} of Φ_{I} (of type $[\mathsf{W}^2] \multimap \mathsf{W}^2$) where W has been replaced by W^2.

Note that there is a *gap* between:

- The length of the evaluation $d : \delta\mathsf{I} \to_{\mathtt{lo}} \mathsf{II} \to_{\mathtt{lo}} \mathsf{I}$, that takes 2 steps, plus the size of the normal form $|\mathsf{I}|_{\mathtt{in}} = 1$, so that $2|d| + |\mathsf{I}|_{\mathtt{in}} = 5$, and
- The size of the composable pair $p = ([[\mathsf{Z}^2] \multimap \mathsf{Z}^2, \mathsf{Z}^2] \multimap \mathsf{Z}^2, [[\mathsf{Z}^2] \multimap \mathsf{Z}^2, \mathsf{Z}^2])$, which is 10.

The point is that the types derived by Ψ_δ and Ψ_{I} are *not* minimal, so their sizes are bigger than $|\Psi_\delta|_{\mathtt{in}}$ and $|\Psi_{\mathsf{I}}|_{\mathtt{in}}$, and do not provide exact bounds for $\delta\mathsf{I}$. For instance, the size of $[[\mathsf{W}^2] \multimap \mathsf{W}^2, \mathsf{W}^2] \multimap \mathsf{W}^2$, which is the type of Ψ_δ, is 6, while $|\Psi_\delta|_{\mathtt{in}} = 2$ – this is an instance of the mentioned gap. Summing up, *minimal types do not compose*, and *composable types do not give exact bounds*.

Out of the Impasse. De Carvalho solves this *cul-de-sac* using composable pairs *up to substitution*. With respect to our example, he considers the composable pair p given by Ψ, but computes the bound using the pair $p' = ([[\mathsf{Z}] \multimap \mathsf{Y}, \mathsf{Z}] \multimap \mathsf{Y}, [[\mathsf{X}] \multimap \mathsf{X}, [\mathsf{X}'] \multimap \mathsf{X}'])$, which is minimal and non-composable. It is obtained by collecting the types of the dry version of Ψ_δ (of type $[[\mathsf{Z}] \multimap \mathsf{Y}, \mathsf{Z}] \multimap \mathsf{Y}$) and the dry version of Ψ_{I} (typing I twice thus having type $[[\mathsf{X}] \multimap \mathsf{X}, [\mathsf{X}'] \multimap \mathsf{X}']$). The last bit is noting that p' is composable *up to the substitution* $\sigma := \{\mathsf{Z} \leftarrow \mathsf{W}^2\}\{\mathsf{Y} \leftarrow \mathsf{W}^2\}\{\mathsf{X} \leftarrow \mathsf{W}^2\}\{\mathsf{X}' \leftarrow \mathsf{W}\}$, since $p'\sigma = p$.

Roughly, for minimal types to compose, they usually have to be expanded, as we have done in the example when substituting Z with W^2. Such an expansion introduces some noise in the measures, so that even minimal composable pairs might not provide exact bounds. De Carvalho's trick is to reverse the expansion, considering composable pairs up to substitution. Our notion of dry derivation makes the expansion reversal technically clean.

Main Result. We can now prove the main result of the paper, namely de Carvalho's exact bounds from composable types.

▶ **Theorem 25** (Exact bounds from composable types). *Let f and f' be normal. If $d : ff' \to^*_{\mathtt{lo}} f''$ and f'' is normal. Then:*

1. Exact bounds*: there exist $\mathsf{L} \in [\![f]\!]$ and $\mathsf{M} \in [\![f']\!]$ such that $2|d| + |f''|_{\mathtt{in}} = |\mathsf{L}| + |\mathsf{M}| + 1$, and L and M are obtained by drying the composable pair induced by a unitary shrinking derivation for ff'.*
2. From types composable up to substitution*: moreover, (L, M) are composable up to substitution, that is, $(\mathsf{L}, \mathsf{M}) \in \mathtt{ShComPairsSub}(f, f')$.*

Proof.

1. By unitary shrinking completeness (Thm. 9), there is a unitary shrinking derivation $\Phi \rhd ff'$ which, by unitary shrinking correctness, satisfies $2|d| + |f''|_{\mathtt{in}} = |\Phi|_{\mathtt{in}}$. The last rule of Φ is an @ rule, that is, Φ has the following shape:

$$\frac{\Phi_f \rhd\vdash f : \mathsf{N} \multimap \mathsf{L}' \qquad \Phi_{f'} \rhd\vdash f' : \mathsf{N}}{\vdash ff' : \mathsf{L}'} @$$

With L' right linear type. Note that $|\Phi|_{\mathtt{in}} = |\Phi_f|_{\mathtt{in}} + |\Phi_{f'}|_{\mathtt{in}} + 1$. By dry representation (Thm. 19.1), there are dry derivations $\Psi_f \rhd\vdash f : \mathsf{L}$ and $\Psi_{f'} \rhd\vdash f' : \mathsf{M}$ such that $\Phi_f \sim \Psi_f$ and $\Phi_{f'} \sim \Psi_{f'}$. By the properties of skeletal invariants (Lemma 15), $|\Phi_f|_{\mathtt{in}} = |\Psi_f|_{\mathtt{in}}$ and $|\Phi_{f'}|_{\mathtt{in}} = |\Psi_{f'}|_{\mathtt{in}}$. By the fact that dry derivations have minimal types (Prop. 17), $|\Psi_f|_{\mathtt{in}} = |\mathsf{L}|$ and $|\Psi_{f'}|_{\mathtt{in}} = |\mathsf{M}|$. Putting it all together, we obtain:

$$\begin{array}{lll} 2|d| + |f''|_{\mathtt{in}} & =_{T.9} & |\Phi_f|_{\mathtt{in}} + |\Phi_{f'}|_{\mathtt{in}} + 1 \\ & =_{L.15} & |\Psi_f|_{\mathtt{in}} + |\Psi_{f'}|_{\mathtt{in}} + 1 \quad =_{Pr.\ 17} \quad |\mathsf{L}| + |\mathsf{M}| + 1 \end{array}$$

2. By the *type substitution* part of the dry representation theorem (Thm. 19.2), there exist substitutions σ_f and $\sigma_{f'}$ such that $\mathsf{L}\sigma_f = \mathsf{N} \multimap \mathsf{L}'$ and $\mathsf{M}\sigma_{f'} = \mathsf{N}$. By stability of dry derivations under renaming (Lemma 18), we can assume that the supports of Ψ_f and $\Psi_{f'}$ are disjoint, so that the domains of σ_f and $\sigma_{f'}$ are disjoint, allowing us to define σ as $\sigma_f \cup \sigma_{f'}$, for which $\mathsf{L}\sigma = \mathsf{L}\sigma_f = \mathsf{N} \multimap \mathsf{L}'$ and $\mathsf{M}\sigma = \mathsf{M}\sigma_{f'} = \mathsf{N}$. Finally, note that L' is right by hypothesis (because Φ is shrinking), so that $(\mathsf{L}, \mathsf{M}) \in \mathtt{ShComPairsSub}(f, f')$. ◀

8 The Less Satisfying Head Case

We conclude our study by adapting the bounds from composable types to the case of head reduction. The study is slightly different in that we omit the study of type substitutions and dry derivations. We do so to show that one can obtain the first part of the de Carvalho's result – which in our opinion is the important one – by resting only on the simpler *size representation theorem* of Sect. 6, with no need of bothering about countably many type variables and dry derivations.

We give only the proof of the main theorem, as the other ones are variants of those in the previous section. They can be found in the technical report [1].

The Head Case. Let $\mathtt{ComPairs}(t, u)$ and $\mathtt{ComPairsSub}(t, u)$ be the analogous sets of $\mathtt{ShComPairs}(t, u)$ and $\mathtt{ShComPairsSub}(t, u)$ but without asking that the composed type is right shrinking, which is not needed for characterizing head termination.

▶ **Lemma 26** (Head normalization and composable types). *Let t and u be closed terms. Then tu $\to_{\mathtt{h}}$-normalizes if and only if $\mathtt{ComPairs}(t, u) \neq \emptyset$.*

The next theorem adapts lax bounds.

▶ **Theorem 27** (Lax bounds for head reduction from composable types). *Let f and f' be closed normal terms such that $d : ff' \to^*_{\mathtt{lo}} h$ with h head normal. Then:*

1. Lax bounds and types*:* $2|d| + |h|_{\mathtt{h}} \leq |\mathsf{L}| + |\mathsf{M}| + 1$ *for every composable pair* $(\mathsf{L}, \mathsf{M}) \in \mathtt{ComPairs}(f, f')$.
2. Lax bounds and types, up to substitutions*:* $2|d| + |h|_{\mathtt{h}} \leq |\mathsf{L}| + |\mathsf{M}| + 1$ *for every composable pair up to substitution* $(\mathsf{L}, \mathsf{M}) \in \mathtt{ComPairsSub}(f, f')$.

▶ **Theorem 28** (Exact bounds for head reduction from composable types). *Let f and f' be normal and such that $d : ff' \to^*_{\mathtt{h}} h$ with h head normal. Then there exist $\mathsf{L} \in [\![f]\!]$ and $\mathsf{M} \in [\![f']\!]$ such that $2|d| + |h|_{\mathtt{h}} = |\mathsf{L}| + |\mathsf{M}| + 1$.*

Proof. By head completeness (Thm. 6), there is a derivation $\Phi \rhd ff'$ satisfying $2|d| + |h|_{\mathtt{h}} = |\Phi|_{\mathtt{in}}$. The last rule of Φ is an @ rule, that is, Φ has the following shape:

$$\dfrac{\Phi_f \rhd\vdash f : \mathsf{N} \multimap \mathsf{L}' \qquad \Phi_{f'} \rhd\vdash f' : \mathsf{N}}{\vdash ff' : \mathsf{L}'}\ @$$

Note that $|\Phi|_{\mathtt{in}} = |\Phi_f|_{\mathtt{in}} + |\Phi_{f'}|_{\mathtt{in}} + 1$. By size representation (Thm. 20), there are $\Psi_f \rhd\vdash f : \mathsf{L}$ and $\Psi_{f'} \rhd\vdash f' : \mathsf{M}$ such that $\Phi_f \sim \Psi_f$ and $|\Psi_f|_{\mathtt{in}} = |\mathsf{L}|$, and $\Phi_{f'} \sim \Psi_{f'}$ and $|\Psi_{f'}|_{\mathtt{in}} = |\mathsf{M}|$.

By the properties of skeletal invariants (Lemma 15), $|\Phi_f|_{\mathtt{in}} = |\Psi_f|_{\mathtt{in}}$ and $|\Phi_{f'}|_{\mathtt{in}} = |\Psi_{f'}|_{\mathtt{in}}$. Putting it all together, we obtain:

$$\begin{array}{lllll} 2|d| + |h|_{\mathtt{h}} & =_{T.6} & |\Phi_f|_{\mathtt{in}} + |\Phi_{f'}|_{\mathtt{in}} + 1 & & \\ & =_{L.15} & |\Psi_f|_{\mathtt{in}} + |\Psi_{f'}|_{\mathtt{in}} + 1 & =_{T.20} & |\mathsf{L}| + |\mathsf{M}| + 1. \end{array}$$ ◀

Note that the hypotheses *f and f' are normal* is not a typo, we do mean *normal*, not *head normal*. The stronger hypotheses are required because the evaluation d leading to h might involve arbitrary sub-terms of f and f', not just their heads. For instance if $f := \lambda x.xt$ and $f' := \mathsf{I}$ then $ff' \to_{\mathtt{h}} \mathsf{I}t \to_{\mathtt{h}} t$, and t is not a head sub-term of f.

This is where the results are less satisfying. Having to assume that f and f' are normal might be acceptable for leftmost reduction but it limits considerably the value of de Carvalho's analysis in the head case, which is the case naturally corresponding to relational semantics. Indeed, there are many head normalizing terms that have no normal form – the paradigmatic example begin given by *fix-point operators* – and about which Thm. 28 does not say anything.

9 Conclusions

In 2007, de Carvalho developed a sharp quantitative analysis of the λ-calculus using multi types and the relational model. His study has been influential, leading to a recent new wave of studies in the λ-calculus halfway between operational and denotational semantics. Only the first and simpler half of de Carvalho's results, however, has really permeated the community. The second more technical – and probably even more interesting – part, which lifts the quantitative analysis to the relational model, has instead been ignored by the recent literature. This paper dissects it and revisits it, pointing out the underlying subtleties and clarifying the concepts and tools for its proof.

A preliminary version of this work led to the adaption of de Carvalho's compositional bounds to call-by-value, which can be found in the technical report [11] by Accattoli et al. Hopefully, further adaptations to other λ-calculi will be developed.

About future work, a sharper study for the head case should be developed, as to avoid the normal form hypotheses. The main weakness of de Carvalho's results, indeed, is that they really work only for strong reduction, at present. As we hinted at in the introduction, probably a finer understanding of the external *vs* internal behaviour of terms is needed.

References

1. Beniamino Accattoli. Semantic bound and multi types, revisited, 2023. arXiv:2311.18233.
2. Beniamino Accattoli and Ugo Dal Lago. (Leftmost-outermost) Beta reduction is invariant, indeed. *Logical Methods in Computer Science*, 12(1), 2016. doi:10.2168/LMCS-12(1:4)2016.
3. Beniamino Accattoli, Ugo Dal Lago, and Gabriele Vanoni. The (in)efficiency of interaction. *Proc. ACM Program. Lang.*, 5(POPL):1–33, 2021. doi:10.1145/3434332.
4. Beniamino Accattoli, Ugo Dal Lago, and Gabriele Vanoni. The space of interaction. In *36th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2021, Rome, Italy, June 29 – July 2, 2021*, pages 1–13. IEEE, 2021. doi:10.1109/LICS52264.2021.9470726.
5. Beniamino Accattoli, Ugo Dal Lago, and Gabriele Vanoni. Multi types and reasonable space. *Proc. ACM Program. Lang.*, 6(ICFP):799–825, 2022. doi:10.1145/3547650.
6. Beniamino Accattoli, Claudia Faggian, and Giulio Guerrieri. Factorization and normalization, essentially. In Anthony Widjaja Lin, editor, *Programming Languages and Systems – 17th Asian Symposium, APLAS 2019, Nusa Dua, Bali, Indonesia, December 1-4, 2019, Proceedings*, volume 11893 of *Lecture Notes in Computer Science*, pages 159–180. Springer, 2019. doi:10.1007/978-3-030-34175-6_9.
7. Beniamino Accattoli, Stéphane Graham-Lengrand, and Delia Kesner. Tight typings and split bounds, fully developed. *J. Funct. Program.*, 30:e14, 2020. doi:10.1017/S095679682000012X.
8. Beniamino Accattoli and Giulio Guerrieri. Types of fireballs. In *Programming Languages and Systems – 16th Asian Symposium, APLAS 2018, Wellington, New Zealand, December 2-6, 2018, Proceedings*, volume 11275 of *Lecture Notes in Computer Science*, pages 45–66. Springer, 2018. doi:10.1007/978-3-030-02768-1_3.

9 Beniamino Accattoli and Giulio Guerrieri. The theory of call-by-value solvability. *Proc. ACM Program. Lang.*, 6(ICFP):855–885, 2022. doi:10.1145/3547652.

10 Beniamino Accattoli, Giulio Guerrieri, and Maico Leberle. Types by need. In *Programming Languages and Systems – 28th European Symposium on Programming, ESOP 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*, volume 11423 of *Lecture Notes in Computer Science*, pages 410–439. Springer, 2019. doi:10.1007/978-3-030-17184-1_15.

11 Beniamino Accattoli, Giulio Guerrieri, and Maico Leberle. Semantic bounds and strong call-by-value normalization. *CoRR*, abs/2104.13979, 2021. URL: https://arxiv.org/abs/2104.13979.

12 Sandra Alves, Delia Kesner, and Miguel Ramos. Quantitative global memory. In Helle Hvid Hansen, Andre Scedrov, and Ruy J. G. B. de Queiroz, editors, *Logic, Language, Information, and Computation – 29th International Workshop, WoLLIC 2023, Halifax, NS, Canada, July 11-14, 2023, Proceedings*, volume 13923 of *Lecture Notes in Computer Science*, pages 53–68. Springer, 2023. doi:10.1007/978-3-031-39784-4_4.

13 Sandra Alves, Delia Kesner, and Daniel Ventura. A quantitative understanding of pattern matching. In *25th International Conference on Types for Proofs and Programs, TYPES 2019, June 11-14, 2019, Oslo, Norway*, volume 175 of *LIPIcs*, pages 3:1–3:36. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.TYPES.2019.3.

14 Alexis Bernadet and Stéphane Lengrand. Non-idempotent intersection types and strong normalisation. *Logical Methods in Computer Science*, 9(4), 2013. doi:10.2168/LMCS-9(4:3)2013.

15 Flavien Breuvart, Giulio Manzonetto, and Domenico Ruoppolo. Relational graph models at work. *Log. Methods Comput. Sci.*, 14(3), 2018. doi:10.23638/LMCS-14(3:2)2018.

16 Antonio Bucciarelli, Alberto Carraro, Thomas Ehrhard, and Giulio Manzonetto. Full abstraction for resource calculus with tests. In Marc Bezem, editor, *Computer Science Logic, 25th International Workshop / 20th Annual Conference of the EACSL, CSL 2011, September 12-15, 2011, Bergen, Norway, Proceedings*, volume 12 of *LIPIcs*, pages 97–111. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2011. doi:10.4230/LIPIcs.CSL.2011.97.

17 Antonio Bucciarelli and Thomas Ehrhard. On phase semantics and denotational semantics: the exponentials. *Ann. Pure Appl. Logic*, 109(3):205–241, 2001. doi:10.1016/S0168-0072(00)00056-7.

18 Antonio Bucciarelli, Thomas Ehrhard, and Giulio Manzonetto. A relational model of a parallel and non-deterministic lambda-calculus. In Sergei N. Artëmov and Anil Nerode, editors, *Logical Foundations of Computer Science, International Symposium, LFCS 2009, Deerfield Beach, FL, USA, January 3-6, 2009. Proceedings*, volume 5407 of *Lecture Notes in Computer Science*, pages 107–121. Springer, 2009. doi:10.1007/978-3-540-92687-0_8.

19 Antonio Bucciarelli, Delia Kesner, Alejandro Ríos, and Andrés Viso. The bang calculus revisited. In *Functional and Logic Programming – 15th International Symposium, FLOPS 2020, Akita, Japan, September 14-16, 2020, Proceedings*, pages 13–32. Springer, 2020. doi:10.1007/978-3-030-59025-3_2.

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

21 Alberto Carraro, Thomas Ehrhard, and Antonino Salibra. Exponentials with infinite multiplicities. In Anuj Dawar and Helmut Veith, editors, *Computer Science Logic, 24th International Workshop, CSL 2010, 19th Annual Conference of the EACSL, Brno, Czech Republic, August 23-27, 2010. Proceedings*, volume 6247 of *Lecture Notes in Computer Science*, pages 170–184. Springer, 2010. doi:10.1007/978-3-642-15205-4_16.

22 Mario Coppo, Mariangiola Dezani-Ciancaglini, and Betti Venneri. Functional characters of solvable terms. *Math. Log. Q.*, 27(2-6):45–58, 1981. doi:10.1002/malq.19810270205.

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

24 Vincent Danos and Thomas Ehrhard. Probabilistic coherence spaces as a model of higher-order probabilistic computation. *Inf. Comput.*, 209(6):966–991, 2011. doi:10.1016/j.ic.2011.02.001.

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

26 Daniel de Carvalho. The relational model is injective for multiplicative exponential linear logic. In Jean-Marc Talbot and Laurent Regnier, editors, *25th EACSL Annual Conference on Computer Science Logic, CSL 2016, August 29 – September 1, 2016, Marseille, France*, volume 62 of *LIPIcs*, pages 41:1–41:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.CSL.2016.41.

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

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

29 Daniel de Carvalho and Lorenzo Tortora de Falco. A semantic account of strong normalization in linear logic. *Inf. Comput.*, 248:104–129, 2016. doi:10.1016/j.ic.2015.12.010.

30 Thomas Ehrhard. Hypercoherences: A strongly stable model of linear logic. *Math. Struct. Comput. Sci.*, 3(4):365–385, 1993. doi:10.1017/S0960129500000281.

31 Thomas Ehrhard. Finiteness spaces. *Math. Struct. Comput. Sci.*, 15(4):615–646, 2005. doi:10.1017/S0960129504004645.

32 Thomas Ehrhard. Collapsing non-idempotent intersection types. In *Computer Science Logic (CSL'12) – 26th International Workshop/21st Annual Conference of the EACSL, CSL 2012, September 3-6, 2012, Fontainebleau, France*, volume 16 of *LIPIcs*, pages 259–273. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2012. doi:10.4230/LIPIcs.CSL.2012.259.

33 Thomas Ehrhard. The scott model of linear logic is the extensional collapse of its relational model. *Theor. Comput. Sci.*, 424:20–45, 2012. doi:10.1016/j.tcs.2011.11.027.

34 Thomas Ehrhard. Non-idempotent intersection types in logical form. In Jean Goubault-Larrecq and Barbara König, editors, *Foundations of Software Science and Computation Structures – 23rd International Conference, FOSSACS 2020, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2020, Dublin, Ireland, April 25-30, 2020, Proceedings*, volume 12077 of *Lecture Notes in Computer Science*, pages 198–216. Springer, 2020. doi:10.1007/978-3-030-45231-5_11.

35 José Espírito Santo, Delia Kesner, and Loïc Peyrot. A faithful and quantitative notion of distant reduction for generalized applications. In Patricia Bouyer and Lutz Schröder, editors, *Foundations of Software Science and Computation Structures – 25th International Conference, FOSSACS 2022, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2022, Munich, Germany, April 2-7, 2022, Proceedings*, volume 13242 of *Lecture Notes in Computer Science*, pages 285–304. Springer, 2022. doi:10.1007/978-3-030-99253-8_15.

36 Jean-Yves Girard. Linear Logic. *Theoretical Computer Science*, 50:1–102, 1987. doi:10.1016/0304-3975(87)90045-4.

37 Jean-Yves Girard. Normal functors, power series and the λ-calculus. *Annals of Pure and Applied Logic*, 37:129–177, 1988. doi:10.1016/0168-0072(88)90025-5.

38 Martin Hyland, Misao Nagayama, John Power, and Giuseppe Rosolini. A category theoretic formulation for engeler-style models of the untyped λ-calculus. *Electronic Notes in Theoretical Computer Science*, 161:43–57, 2006. Proceedings of the Third Irish Conference on the Mathematical Foundations of Computer Science and Information Technology (MFCSIT 2004). doi:10.1016/j.entcs.2006.04.024.

39 Delia Kesner, Loïc Peyrot, and Daniel Ventura. The spirit of node replication. In Stefan Kiefer and Christine Tasson, editors, *Foundations of Software Science and Computation Structures – 24th International Conference, FOSSACS 2021, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2021, Luxembourg City, Luxembourg, March 27 – April 1, 2021, Proceedings*, volume 12650 of *Lecture Notes in Computer Science*, pages 344–364. Springer, 2021. doi:10.1007/978-3-030-71995-1_18.

40 Delia Kesner and Daniel Ventura. Quantitative types for the linear substitution calculus. In Josep Díaz, Ivan Lanese, and Davide Sangiorgi, editors, *Theoretical Computer Science – 8th IFIP TC 1/WG 2.2 International Conference, TCS 2014, Rome, Italy, September 1-3, 2014. Proceedings*, volume 8705 of *Lecture Notes in Computer Science*, pages 296–310. Springer, 2014. doi:10.1007/978-3-662-44602-7_23.

41 Delia Kesner and Pierre Vial. 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*, pages 619–632, 2020. doi:10.1145/3373718.3394774.

42 Delia Kesner and Andrés Viso. Encoding tight typing in a unified framework. In Florin Manea and Alex Simpson, editors, *30th EACSL Annual Conference on Computer Science Logic, CSL 2022, February 14-19, 2022, Göttingen, Germany (Virtual Conference)*, volume 216 of *LIPIcs*, pages 27:1–27:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.CSL.2022.27.

43 A. J. Kfoury and J. B. Wells. Principality and type inference for intersection types using expansion variables. *Theor. Comput. Sci.*, 311(1-3):1–70, 2004. doi:10.1016/j.tcs.2003.10.032.

44 Jean-Louis Krivine. *Lambda-calculus, types and models.* Ellis Horwood series. Masson, 1993.

45 Jim Laird, Giulio Manzonetto, Guy McCusker, and Michele Pagani. 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*, pages 301–310. IEEE Computer Society, 2013. doi:10.1109/LICS.2013.36.

46 Giulio Manzonetto. A general class of models of H^*. In Rastislav Královic and Damian Niwinski, editors, *Mathematical Foundations of Computer Science 2009, 34th International Symposium, MFCS 2009, Novy Smokovec, High Tatras, Slovakia, August 24-28, 2009. Proceedings*, volume 5734 of *Lecture Notes in Computer Science*, pages 574–586. Springer, 2009. doi:10.1007/978-3-642-03816-7_49.

47 Giulio Manzonetto and Domenico Ruoppolo. Relational graph models, taylor expansion and extensionality. In Bart Jacobs, Alexandra Silva, and Sam Staton, editors, *Proceedings of the 30th Conference on the Mathematical Foundations of Programming Semantics, MFPS 2014, Ithaca, NY, USA, June 12-15, 2014*, volume 308 of *Electronic Notes in Theoretical Computer Science*, pages 245–272. Elsevier, 2014. doi:10.1016/j.entcs.2014.10.014.

48 C.-H. Luke Ong. 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*, pages 1–12. IEEE Computer Society, 2017. doi:10.1109/LICS.2017.8005064.

49 Luca Paolini, Mauro Piccolo, and Simona Ronchi Della Rocca. Essential and relational models. *Math. Struct. Comput. Sci.*, 27(5):626–650, 2017. doi:10.1017/S0960129515000316.

50 Simona Ronchi Della Rocca. Principal type scheme and unification for intersection type discipline. *Theor. Comput. Sci.*, 59:181–209, 1988. doi:10.1016/0304-3975(88)90101-6.