Quantitative Semantics of the Lambda Calculus: Some Generalisations of the Relational Model

C.-H. Luke Ong
University of Oxford

Abstract—We present an overview of some recent work on the quantitative semantics of the \( \lambda \)-calculus. Our starting point is the fundamental degenerate model of linear logic, the relational model. We show that three quantitative semantics of the simply-typed \( \lambda \)-calculus are equivalent: the relational semantics, HO/N game semantics, and the Taylor expansion semantics. We then consider two recent generalisations of the relational model: first, \( R \)-weighted relational models where \( R \) is a complete commutative semiring, as studied by Laird et al.; secondly, generalised species of structures, as introduced by Fiore et al. In each case, we briefly discuss some applications to quantitative analysis of higher-order programs.

I. AN OVERVIEW OF QUANTITATIVE MODELS

Denotational semantics (or Scott-Strachey semantics) [73] is an approach to formalising the meanings of programs by mapping them into some abstract domain of mathematical objects. A central tenet of denotational semantics is that the semantics should be defined compositionally. The theory of equality induced by the denotational semantics gives a natural notion of program equivalence: two programs are equal if they have the same denotation. On the other hand, according to operational semantics, the meaning of a program is the behaviour of some (typically abstract) machine when running it. There is a compelling notion of program equivalence based on operational semantics: two program phrases are observationally equivalent just if they compute the same outcome when one is replaced by the other in all possible program contexts. A central problem in the semantics of programming languages is to construct, for a given class of languages, a denotational semantics that agrees with the operational semantics. The strongest such goodness-of-fit criterion is full abstraction [62,70]: the coincidence of the denotational equality of programs with observational equivalence.

Though the basis of a successful programming language theory, the notion of observational equivalence, and the related developments in denotational and operational semantics, have tended to ignore quantitative notions such as time, space and energy as computational resource, or in the case of nondeterministic and probabilistic computation, such quantities as the probability of a successful computation, and the expected termination time.

A major step in quantitative semantics was Girard’s linear logic [39]. A refinement of classical and intuitionistic logic, linear logic emphasises the rôle of formulas as resources. The use of linear logic as an organisational principle in semantics of computation has the advantage of immediately revealing information about resource usage. Unsurprisingly models of linear logic are quantitative. This is already evident in the simplest degenerate model, the relational model [13,39], in which multisets are used to record the number of times a resource is used.

Around the time of the introduction of linear logic [39], Girard also proposed the normal functor semantics of the \( \lambda \)-calculus [40]. In this semantics, a term is interpreted as a formal power series with set-valued coefficients, i.e., as a functor \( X \to \text{Set} \), where the set \( X \) is the denotation of a data type. In that work, Girard was “mainly concerned with a quantitative approach: not only to say when \( f \) takes the value \( \cdots \) at argument \( \cdots \), but also how many times it does” [40, p. 172]. His original intuitions came from linear algebra: data types are interpreted as vector spaces; a resource of type \( A \) is a vector \( \sum_{a \in \text{base}(A)} m_a \cdot a \) where the coefficient \( m_a \) gives the multiplicity of the atomic datum \( a \) of type \( A \) in the resource. Programs are interpreted as power series or analytic functions; programs that use their input exactly once then correspond to linear functions. The idea is that, by choosing the appropriate coefficients, we should be able to use such a semantics to analyse programs, not only qualitatively, with respect to “what they can do”, but also quantitatively, with respect to “in how many steps”, or “in how many different ways”, or “with what probability”.

Girard’s work [39,40] engendered a rich vein of research, initially in models of linear logic. In his normal functor model, the scalars are sets. Ehrhard’s Kôthe sequence spaces [23] and finiteness spaces [24] recast Girard’s intuitions on actual vector spaces over fields (the former over \( \mathbb{R} \) and \( \mathbb{C} \), and the latter over any field). In the weighted relational models of Lamarche [56] and Laird et al. [51,55], the scalars are elements of any continuous commutative semiring, or, more generally, any complete commutative semiring. In the quantitative model of a higher-order quantum programming language studied by Pagani et al. [67], the scalars are completely positive maps of a finite dimension. Girard has suggested how his coherence spaces model [38,39] of linear logic can be refined to give an account of probabilistic computation. Building on Girard’s ideas, Danos, Ehrhard et al. [18,31] have analysed probabilistic coherence spaces as a model of higher-order probabilistic computation.

In (idealised) quantitative semantics, programs are analytic functions, which are infinitely differentiable. This motivates the question of understanding differentiation as a program-
ming construct in a higher-order setting; and it led Ehrhard and Regnier to introduce the differential λ-calculus [27], a differential calculus for higher-order functions, and an accompanying differential linear logic [29]. In a follow-up paper [30], Ehrhard and Regnier introduced the Taylor expansion of a λ-term as a formal sum (with rational coefficients) of terms of the resource calculus [30, 69], which may be viewed as linear approximants of the λ-term. Models of the differential λ-calculus (and resource calculus) [6, 14] are naturally quantitative. Many models of linear logic give rise to models of the differential λ-calculus / linear logic. Particularly attractive is the category CVS of convenient vector spaces (C∞-complete locally convex topological vector spaces) and bornological linear maps [7], in the sense of Frölicher and Kriegl [36]. The category CVS supports a linear exponential comonad for which the Kleisli category is the category of smooth maps on convenient vector spaces.

There are proposals, from the programming language community, of denotational and operational semantics that carry quantitative information. Game semantics [1, 44, 63] is a denotational semantics with a strong operational flavour that typically captures more intensional information about the computation (such as the number of times an input argument is evaluated) than the more abstract Scott-Strachey style denotational semantics. Exploiting the quantitative nature of HO/N game model [44], Férée [32] has recently proposed a definition of cost based on a refined notion of program equivalence. Inspired by Sand’s ideas, Ghica has given a game semantics, called slot games [37], which is induced by a notion of observation formalised in Sand’s theory of improvement.

Outline of the paper

In this survey paper, we start from the relational model, and consider two ways to generalise it.

In Section II we introduce MRel, the Kleisli category of the finite-multiset comonad on the category Rel of sets and relations. We then study three quantitative semantics of the simply-typed λ-calculus: (i) relational semantics (ii) HO/N game semantics, and (iii) Taylor expansion semantics. We show that they are equivalent in an appropriate sense.

In Section III we consider the first generalisation of the relational model, namely, the weighted relational models, where the weights are elements of any continuous commutative semiring, and, more generally, any complete commutative semiring. We briefly discuss some applications of these models to quantitative analysis of nondeterministic higher-order computation.

Section IV concerns the second generalisation of the relational model, which is in a 2-categorical direction. We introduce the bicategory Prof of profunctors, and the cartesian closed bicategory ESP of generalised species of structures. We analyse the ESP-semantics of the nondeterministic λY-calculus, and show that it coincides with the rigid Taylor expansion semantics. Rigid Taylor expansion of a λ-term is a version of Taylor expansion that uses linear approximants in the form of rigid resource terms, which are list-based (as opposed to bag-based), and considered modulo an isomorphism action.

II. THE RELATIONAL MODEL AND ITS VARIOUS FACES

The relational model [40] is the simplest degenerate model of the linear logic. It underlies most denotational models of (differential) linear logic [27, 39], and serves to motivate the general constructions of quantitative models in Sections III and IV. The relational semantics of the λ-calculus is equivalent to type assignment in a system of commutative, associative and nonidempotent, intersection refinement types. We relate the relational semantics to HO/N game semantics on the one hand, and to Taylor expansion semantics on the other.

A. The relational model MRel

We start from the category Rel of sets and relations. Given sets A and B, Rel(A, B) := P(A × B). The identities are the diagonal relations: idA := {(a, a) | a ∈ A}; and the composite of s ∈ Rel(A, B) and t ∈ Rel(B, C) is just relational composition:

(s; t) := {(a, c) ∈ A × C | ∃b ∈ B. (a, b) ∈ s ∧ (b, c) ∈ t}.

Given sets A and B, their tensor product A ⊗ B = A × B is the cartesian product, with unit 1 = {∗}, an arbitrary singleton set. Rel is *-autonomous: the linear function space A → B = A × B, with the natural bijection Rel(C ⊗ A, B) ≃ Rel(C, A ⊗ B) being induced by the cartesian product associativity isomorphism. The categorical product A1 ∩ A2 is the disjoint union, and the terminal object ∅ = ∅. The dualising object ⊥ = 1, and so, Rel is compact closed.

Notation: We represent a finite multiset m over a set A as an unordered list [a1, · · · , an], and say that n is its cardinality. We write the union of multisets m and m′ as m + m′, and Mfin(A) for the set of finite multisets over A.

The finite-multiset construction, Mfin, is a comonad on Rel, acting on morphisms s ∈ Rel(A, B) as Mfin(s) := {[m1, · · · , mn] : [b1, · · · , bn]} with unit delA := {[a] : Mfin(A) → A} : Mfin(A) → A and multiplication delc := {[θ1 + · · · + θn, θ1, · · · , θn]} | ∀i ≤ n. θi ∈ Mfin(A)} : Mfin(A) → Mfin(A).

We define the category MRel as the Kleisli category of the Mfin comonad. It is useful to give a direct description of MRel.

- The objects of MRel are sets.
- A morphism from A to B is a relation from Mfin(A) to B. I.e. MRel(A, B) := P(Mfin(A) × B).

The identity map of A is the relation idA := {[a] : a ∈ A}.

The composite of s ∈ MRel(A, B) and t ∈ MRel(B, C) is

(s; t) := {(m, c) | ∃(m1, b1), · · · , (mk, bk) ∈ s.

m = m1 + · · · + mk ∧ (b1, · · · , bk), c ∈ t}.

MRel is cartesian closed. The products in Rel give the products in MRel. It is convenient to regard the canonical
bijection $\mathcal{M}_{\text{fin}}(A_1) \times \mathcal{M}_{\text{fin}}(A_2) \cong \mathcal{M}_{\text{fin}}(A_1 + A_2)$ as equality. Thus we will still write $(m_1, m_2)$ for the corresponding element of $\mathcal{M}_{\text{fin}}(A_1 + A_2)$. Given sets $A$ and $B$, the exponential object $B^A$ is $\mathcal{M}_{\text{fin}}(A) \times B$, and the evaluation morphism $\text{ev}_{A,B} \in \text{MRel}(B^A \cap A, B)$ is

$$\text{ev}_{A,B} := \{((m, b), m, b) \mid m \in \mathcal{M}_{\text{fin}}(A), b \in B\}$$

Given $s \in \text{MRel}(C \cap A, B)$, its exponential transpose $\lambda(s) := \{((m, (m', b)) \mid (m, m', b) \in s\} \in \text{MRel}(C, B^A)$.

$\text{MRel}$ is a differential $\lambda$-category [6, 14]: the homsets are endowed with the semi-additive structure $(\text{MRel}(A, B), +, \emptyset)$. Given $s \in \text{MRel}(A, B)$, we define its derivative $D(s)$ in $\text{MRel}(A \cap A, B)$ as

$$D(s) := \{((a, m), b) \mid (m, [a], b) \in s\}.$$  

B. Relational semantics as refinement type assignment

Simple types are defined by the grammar: $A, B ::= \alpha \mid A \rightarrow A$, where $\alpha$ is the unique atomic type. The relational semantics of a $\lambda$-term-in-context $\Gamma \vdash M : A$, written $\Gamma \vdash M : A^{\text{MRel}}$, is determined by the cartesian closed structure of $\text{MRel}$, once the interpretation of the atomic type $\alpha$ is given. Let $\mathcal{X}$ be a fixed set which is assumed to be countably infinite. (The assumption is not necessary for the relational semantics to be well-defined; see Remark 8.) Define $[A]^{\text{MRel}}$ inductively by

$$[\emptyset]^{\text{MRel}} := \mathcal{X},$$

$$[A \rightarrow B]^{\text{MRel}} := \mathcal{M}_{\text{fin}}([A]^{\text{MRel}}) \times [B]^{\text{MRel}}.$$  

Let $\Gamma = x_1 : A_1, \ldots, x_n : A_n$. The relational semantics $\Gamma \vdash M : A$ is then a subset of

$$(\mathcal{M}_{\text{fin}}([A_1]^{\text{MRel}}) \times \cdots \times \mathcal{M}_{\text{fin}}([A_n]^{\text{MRel}})) \times [A]^{\text{MRel}}.$$  

Refinement types: A simple but important observation is that the relational semantics $[A]^{\text{MRel}}$ of a simple type $A$ can be seen as the set of all nondempotent intersection types that refine $A$ [12, 20]. Let $\alpha$ and $\beta$ be elements of $\mathcal{X}$. Then an element of $[\alpha]^{\text{MRel}}$ is an atomic type $\alpha \in \mathcal{X}$. Given a sequence $a_1, \ldots, a_n$ of elements in $[A]^{\text{MRel}}$, we write $a_1 \wedge \cdots \wedge a_n$ to mean the multiset $[a_1, \ldots, a_n] \in \mathcal{M}_{\text{fin}}([A]^{\text{MRel}})$. It follows that the intersection connective $\wedge$ is commutative, associative and nondempotent, as in Kfoury’s treatment [49] and more recently, de Carvalho’s [20]. An element of $[A \rightarrow B]^{\text{MRel}} = \mathcal{M}_{\text{fin}}([A]^{\text{MRel}}) \times [B]^{\text{MRel}}$ is then a pair $(\theta, a)$, which we write as $\theta \sim a$.

Thus, given a simple type $A$, $[A]^{\text{MRel}}$ can be seen as the set of intersection types defined by the grammar

$$a, b ::= \alpha \mid a \rightarrow a \mid \theta \sim a \mid a_1 \wedge \cdots \wedge a_n \quad (n \geq 0)$$

that refines $A$ (written $a < A$), where the refinement relation $a < A$ and $\theta < !A$ is defined by the following rules.

$$\theta < !A \quad b < B \quad \forall i < n. a_i < A$$

We write $\top$ for the empty intersection; $\sim$ associates to the right, and $\wedge$ takes precedence over $\sim$. In case $a < A$, we say $a$ is a refinement type of $A$; similarly in case $\theta < !A$, we say $\theta$ is a refinement intersection of $A$.

A refinement type judgement is a triple of the form $\Delta \vdash M : b$ where $\Delta$ is an environment, which is defined to be a finite multiset of type bindings of the form $x : a$ such that $a$ is a refinement of the simple type of $x$. Given environments $\Delta_1$ and $\Delta_2$, we write $\Delta_1, \Delta_2$ for their multisets union, and define $\text{dom}(\Delta) := \{x \mid \exists a. (x : a) \in \Delta\}$, the domain of $\Delta$. Figure 1 lists the typing rules.

The refinement type system is equivalent to the relational semantics in the following sense. Take a $\lambda$-term-in-context $\Gamma \vdash M : A$ with $\Gamma = x_1 : A_1, \ldots, x_n : A_n$, and a refinement-type environment $\Delta$ such that $\text{dom}(\Delta) \subseteq \text{dom}(\Gamma)$. Suppose $\Delta = [x_1 : a_{11}, \ldots, x_1 : a_{1r_1}] + \cdots + [x_n : a_{nr_n}, \ldots, x_n : a_{nr_n}]$; write $\Sigma = (a_{11} \wedge \cdots \wedge a_{1r_1}, \ldots, a_{nr_n} \wedge \cdots \wedge a_{nr_n})$. Then we have:

**Theorem 1.** $\Delta \vdash M : a$ iff $(\Sigma, a) \in \Gamma \vdash M : A^{\text{MRel}}$.

Thus the relational semantics and refinement type assignment are equivalent, and we shall use them interchangeably.

C. Playful types are the inhabited refinement types

First a quick review of the syntax of the resource $\lambda$-calculus [11, 25, 30]. Resource terms and bags are given by the grammar:

$$M, N ::= x | \lambda x. M | M \cdot P$$

$$P, Q ::= [M_1, \ldots, M_n] \quad (n \geq 0).$$

We call $M$ a term, and call $P$, which is a finite multiset of terms, a bag. Since a bag is a multiset, it is identified with a permutation of its elements; as is standard, $\alpha$-equivalent terms (respectively bags) are identified. Henceforth we shall only consider simply-typed resource terms and bags (which means that all terms in a bag must have the same simple type). A typed term is $\beta$-normal if it does not have a subterm of the form $(\lambda x. M)P$; it is $\eta$-long if every application and variable in the term is fully applied. A resource term is normal if it is $\beta$-normal and $\eta$-long.

We define a refinement type assignment system for resource $\lambda$-terms. The typing rules are listed in Fig. 2 (except for the last, they are the same as the rules in Fig. 1). We make the same assumptions about environments $\Delta$ as in the system for $\lambda$-terms. It follows that every provable judgement respects the simple types of the terms and bags.
The induced tree $T_a$ follows.

**Example 3.** Consider the (playful) refinement type $a \triangleleft A$ where $A = (\alpha \rightarrow \text{o}) \rightarrow \text{o}$ and

$$a = \left( (\delta \rightarrow \beta) \land (\top \rightarrow \gamma) \right) \rightarrow \alpha \land (\top \rightarrow \delta) \rightarrow \beta \land (\top \rightarrow \gamma) \rightarrow \alpha.$$

The induced tree $T_a$ (with edges indicated by straight arrows, either solid or dotted) and graph $G_a$ (with edges as solid arrows, either straight or curved) are depicted as follows.

**Theorem 4.** A refinement type is inhabited iff it is playful.

A proof can be found in [78].

**Remark 5.** The original intuition behind the playfulness condition came from (P-visibility in) HO/N game semantics. However an involutive refinement type induces a proof structure, and playfulness is closely connected to the correctness criterion [19, 39] of multiplicative linear logic.

### D. MRel, HO/N games and resource λ-calculus

We show that the following quantitative semantics of the simply-typed λ-calculus are equivalent [78]:

1) Relational semantics (or refinement type assignment)
2) HO/N game semantics [44, 63]
3) Taylor expansion semantics [30]

Take a λ-term $M$. In the relational semantics, $M$ is interpreted as a collection of refinement types. In HO/N game semantics, $M$ is interpreted as a collection of plays. In the Taylor expansion semantics, $M$ is interpreted as a collection of resource terms in normal form.

Underpinning the equivalence of the three semantics is the following technical result [78]:

**Theorem 6.** There exists a family of bijections parametrised by simple types $A$

$$\varpi_A : \{ \text{plays of arena } A \} / \sim \rightarrow \{ \text{normal resource terms of type } A \}$$

that preserves composition., where $\sim$ is the alternating homotopy relation of Melliès [59, 77]. Via the bijections, the game semantics of a λ-term coincides with its Taylor expansion followed by normalisation.

As a consequence, we can define a functor from the category of HO/N games to the Kleisli category MRel.

**Corollary 7.** The category of HO/N games is isomorphic to a sub-cartesian closed category of the Kleisli category MRel.

**Remark 8.** Laird et al. [54] developed a method to construct a differential category [6] from a symmetric monoidal category, and then reconstructed a known category of games from a new category of exhausting games. It follows from this construction that there is a functor from a category of games to the relational model. Compared to our work, this functor can be...
seen as the “colourless” version (i.e. the set of atomic types $\mathcal{X}$ is singleton) of the functor of Corollary 7.

In the rest of this section, we explain the bijective correspondence between the following collections (and direct the reader to [78] for the other details):

1) playful refinement types
2) ($\sim$-equivalence classes of) plays
3) normal resource terms

(Because of the length restriction, we will assume basic knowledge of HO/N games, and refer the reader to [44,77].) A rough and informal sketch of the idea is depicted in Figure 3. A resource term in normal form can be written as a tree, as in the middle of Figure 3. Each node is labelled by $\lambda$-abstraction followed by a head variable.\(^1\) The edges express the relationship between functions and arguments: the child of a node is an argument of the head variable of the parent. Then we line up the nodes of the tree in such a way that every node is located to the left of its children. The resulting sequence of $\lambda$-abstractions and variables is equipped with leftward pointers: the pointer from an abstraction (solid lines in Figure 3) comes from the parent-child relation in the tree, and the pointer from a variable (dotted lines) is determined by the binder-bindee relation. The sequence with pointers is reminiscent of a play and a resource term and prove their uniqueness. See [78] for the details.

A resource term generates a set of plays because the process of lining up is nondeterministic. The main theorem states that the set of plays generated by a resource term is a $\sim$-equivalence class. Moreover every play is generated by a resource term.

The idea should now be intuitively clear. However, to define the bijection $\varpi_A$ of Theorem 6, it is instructive to use the refinement type assignment (or, equivalently, the relational model) as a bridge, because refinement type assignment systems for both the resource calculus and the game model have already been studied: the relational model is a common tool for studying resource terms [11], and game semantics for an intersection type assignment system has been studied [66].

The definition of the bijection $\varpi_A$ of Theorem 6 is illustrated in Figure 4, and is divided into four steps. The simple type in question is $A = \{o_{111} \rightarrow o_{11} \rightarrow o_1 \}$. (We use subscripts to distinguish the occurrences of the atomic type $o$ in $A$.)

Step 1: Colouring a play: We assign a “colour” (written as $\alpha, \beta, \gamma$ and $\delta$ in Fig. 4) to each occurrence of moves in such a way that

- every pair of consecutive O-P moves have the same colour, and
- different occurrences of O-moves (respectively P-moves) have different colours.

\(^1\)We consider $\lambda$-abstraction that can bind a (possibly empty) sequence of variables, although only sequences of length 1 appear in Figure 3.

Thus one needs $n$ colours to annotate a play of length $2n$.

Step 2: Representing a coloured play by a tree: This step simply forgets the sequential structure of the (coloured) play. The resulting structure is a tree whose edge is a justification pointer of the play and whose node is labelled by a pair of a move and a colour. For example, the move and colour of the node named $l_7$ in Figure 4 is $o_{11} \gamma$ and $\delta$. The node named $l_i$ corresponds to the $i$th move in the original play. This structure is called a valued thick subtree in [9], a high-level arena in [66], and a partitioned position in [21].

Step 3: Constructing a refinement type: High-level arenas (or, valued thick subtrees) bijectively correspond to intersection types that refine the simple type. Let us write $a_i$ for the type corresponding to the subtree rooted at $l_i$. For example

(i) $a_8$ is an atomic type $\delta$ that refines $o_{111}$.
(ii) $a_3$ is the type $\delta \sim \beta$ that refines $o_{111} \rightarrow o_{11}$.
(iii) $a_5$ is the type $\top \sim \gamma$ that refines $o_{111} \rightarrow o_{11}$, where $\top$ is the empty intersection type, and
(iv) $a_2$ is the type $(\delta \sim \beta) \land (\top \sim \gamma)$ that refines $(o_{111} \rightarrow o_{11}) \rightarrow o_1$.

The type $a_1 = (a_2 \lor a_3 \land a_6) \rightarrow o$ is written in Figure 4.

Step 4: Computing the inhabitant: The resource term corresponding to the play is the inhabitant of the refinement type. An inhabitant always exists uniquely (up to $\alpha$-conversion) for an refinement type constructed in this way.

There are several difficulties in proving bijectivity of the map defined above. In particular, in the last step, the existence and uniqueness of an inhabitant are challenging to establish. Our strategy is to study the map defined by Steps 1-3 and the inverse of Step 4. We characterise the images of those maps, using game semantics, and show their coincidence.

Then, given a refinement type in the image, we construct a play and a resource term and prove their uniqueness. See [78] for the details.

Preservation of composition by the bijection is relatively easy to prove by applying known results.

E. Taylor expansion as game semantics

For a term $L$ of the simply-typed nondeterministic $\lambda$-calculus, we define $L^*$ by:

\[ x^* := \{ x \} \quad (\lambda x. L)^* := \{ \lambda x. M \mid M \in L^* \} \]
\[ (L L')^* := \{ [N_1, \ldots, N_k] \mid M \in L^*, k \geq 0, \forall i \leq k. N_i \in (L')^* \} \]
\[ (L_1 + L_2)^* := L_1^* \cup L_2^* \]

We call $L^*$ the Taylor expansion of $L$. For example, $((\lambda f. f(\lambda x. f(\lambda y. y)))^*$ contains resource terms $\lambda f. [f[\] and $\lambda f. [\lambda x. f(\lambda y. y)]$ (and others). Let $M$ be a set of resource terms; we write $\text{NF}(M)$ for the set of normal forms of elements of $M$. We write $\llbracket \Gamma \vdash M : A \rrbracket^\varpi$ for the game semantics of a simply-typed term-in-context $\Gamma \vdash M : A$. As an application of Theorem 6, we have the following result [78].

Theorem 9. For every $\eta$-long nondeterministic $\lambda$-term-in-context $\Gamma \vdash M : A$, we have $\varpi(\llbracket \Gamma \vdash M : A \rrbracket^\varpi) = \text{NF}(M^*)$. 

III. Weighted relational models

In this section, we generalise the relational model with weights. The category of weighted relations over a complete commutative semiring \( R \) (equivalently, free \( R \)-semimodules and linear functions) was introduced as a model of linear logic by Lamarche [56]. This model was further developed by Laird et al. (see [51, 55] among other) and its computational properties analysed via a semantics of \( R \)-weighted PCF.

A. Lafont category: a model of intuitionistic linear logic

We begin with a categorical description of models of linear logic as studied in Lafont’s PhD thesis [50]. There are more general definitions, but Lafont’s simple formulation suits our purpose. For a systematic analysis of categorical semantics of linear logic, see Melliès’ monograph [60]. Also relevant is Ehrhard’s account [26] from a differential linear logic perspective.

Recall that an object \( A \) of a symmetric monoidal category (SMC) \( (C, \otimes, 1) \) is a commutative comonoid if it is equipped with a multiplication morphism \( c : A \to A \otimes A \), and a unit morphism \( w : 1 \to A \), such that the usual commutativity, associativity, and unit diagrams commute. A comonoid morphism \( f \) from \( (A, c, w) \) to \( (A', c', w') \) is given by a morphism \( f \in C(A, A') \) satisfying \( f' c = c (\varphi \otimes \varphi) \), and \( f' w = w \). Given a SMC \( (C, \otimes, 1) \), the category \( \text{Comm}(C) \) has commutative comonoids of \( C \) as objects, and comonoid morphisms as morphisms.

**Definition 10** (Lafont category). A symmetric monoidal closed category \( (C, \otimes, \langle - , - \rangle, 0, 1) \) is a Lafont category just if

(i) \( C \) has finite products: the SMC \( (C, \times, \top) \) is cartesian,

(ii) \( C \) has free linear exponentials: the forgetful functor \( U \) from \( \text{Comon}(C) \) to \( C \) has a right adjoint \( ! \).

Condition (ii) above says \( C \) has all free commutative comonoids. That is to say, for every \( A \), there is an object \(!A!\)—the free commutative comonoid generated by \( A \)—endowed with a commutative comonoid structure.

\[
\text{ctr}_A : !A \to !A \otimes !A \quad \text{wk}_A : !A \to 1
\]

and a morphism \( \text{der}_A : !A \to A \) satisfying the universal property: for every comutative comonoid \( B \), and every morphism \( f : B \to A \), there is a unique comonoid morphism \( f! : B \to !A \) satisfying \( f! \circ \text{der}_A = f \). The multiplication (called contraction) and unit (call weakening) morphisms of \(!A\) are named after the standard structural rules in proof theory; and \( \text{der}_A \) is called delocation in the language of linear logic.

We describe the comonad \( (!, \text{der}, \text{dig}) \) induced by the (monoidal) adjunction \( U \dashv ! \) in full. The endofunctor \( ! \) maps a morphism \( f : A \to B \) to \( (\text{der}_A : f)! : !A \to !B \). The multiplication morphism, called digging, is \( \text{dig}_A = (\text{id}_A)! : !A \to !!!A \). The unit morphism is just delocation \( \text{der}_A : !A \to A \). We call the induced comonad the free linear exponential of the Lafont category. Furthermore, setting the mediating morphisms

\[
m^0_\top : 1 \to !\top \quad m^0_{A,B} : !A \otimes !B \to !(A \times B)
\]

as \( m^0_\top = (\top_1)! \) and \( m^2_{A,B} = (\text{der}_A \otimes \text{wk}_B, \text{wk}_A \otimes \text{der}_B)! \), which are natural isomorphisms, makes

\[
(!, m^0, m^2) : (C, \otimes, 1) \to (C, \times, \top)
\]

a strong symmetric monoidal functor. The natural isomorphisms \( m^0 \) and \( m^2 \) are called Seely isomorphisms [39].
The Kleisli category $C_!$ over the comonad $(!, \text{der}, \text{dig})$ is automatically cartesian closed. The products in $C_!$ give the products in $C$, and the intuitionistic function space $A \Rightarrow B$ is $!A \rightarrow B$, which is often called the Girard translation [39]. The existence of an adjunction $(\cdot) \times A \vdash A \Rightarrow (\cdot)$ is a consequence of the symmetric monoidal structure of $C$ and the Seely isomorphisms:

\[
C((C \times A), B) = C((C \times A) \rightarrow B) \\
\cong C((C \odot A), B) \\
\cong C((C, A \rightarrow B) = C(C, !A \rightarrow B)
\]

**Remark 11.** The linear exponentials of a Lafont category are free constructs, by definition. The relational and weighted relational models considered in Sections II and III are Lafont categories, but there are models of linear logic where several relational models considered in Sections II and III are Lafont categories, but there are models of linear logic where several linear exponential modalities may coexist (for example, coherence and hypercoherence spaces models [38, 58], game models [58], and the bicategory of profunctors [16]). To account for this more general setting, Melliès has proposed new-Lafont categories [60], which is a symmetric monoidal closed category $C$ with finite products such that the (restriction to $M$) forgetful functor $U : M \rightarrow C$ has a right adjoint, where $M$ is a full subcategory of $\text{Comon}(C)$ closed under tensor and containing the unit comonoid 1.

**B. Constructing free linear exponentials from SMC**

We discuss a folklore result: a symmetric monoidal category $C$ has free linear exponentials if it has countable distributive biproducts, and all symmetric tensor powers. In the following we write $A^{\odot n}$ for the $n$th tensor power of $A$, i.e., $A^{\odot 0} := 1$ and $A^{\odot (n+1)} := A^{\odot n} \otimes A$.

**Definition 12** (Lafont linear exponential). A family of objects $\{A^n \mid n \in \omega\}$ of a symmetric monoidal category are the symmetric tensor powers of $A$ if

(i) for each $n \geq 0$, the set of $n!$ permutation automorphisms on $A^{\odot n}$ has an equaliser $\text{eq}_n : A^{\odot n} \rightarrow A^{\odot n}$, and

(ii) these equalisers are preserved by the tensor, i.e., for all $m, n \geq 0$

\[
\text{eq}_m \otimes \text{eq}_n : A^{\odot m} \otimes A^{\odot n} \rightarrow A^{\odot m+n}
\]

is an equaliser for the tensor product of pairs of permutation automorphisms.

We say that an object $!A$ of a SMC with biproducts is the Lafont linear exponential of $A$ if it is the biproduct of all the symmetric tensor powers of $A$, i.e., $!A = \bigoplus_{n \in \omega} A^n$.

The Lafont linear exponential $!A$ is endowed with a commutative cocomonoid structure with weakening and contraction morphisms as follows

\[
\text{wk}_A = \pi_0 : !A \rightarrow 1 \\
\text{ctr}_A = (\sigma_m \cdot \sigma_n \mid m, n \in \omega) : !A \rightarrow !A \otimes !A
\]

where $\sigma_{m,n} : A^{[m+n]} \rightarrow A^{[m]} \otimes A^{[n]}$ is the unique morphism satisfying $\text{eq}_{m+n} = \sigma_{m,n} \cdot (\text{eq}_m \otimes \text{eq}_n)$, and $\pi_n : !A \rightarrow A^{[n]}$ is the $n$th projection.

**Proposition 13** ([61]). Let $C$ be a SMC with countable distributive biproducts. If $C$ has Lafont linear exponentials, then they are the free linear exponentials.

**C. Constructing Lafont categories from complete semirings**

Consider $\text{Rel}$, a simple example of Lafont category. Morphisms from $A$ to $B$ may be viewed as $(A \times B)$-matrices over the Boolean semiring $\mathbb{B}$, and the composite of $s \in \text{Rel}(A, B)$ and $t \in \text{Rel}(B, C)$ is just matrix multiplication: the $(a, c)$-entry of the composite $(s; t)$ is the $B$-indexed sum

\[
(s; t)(a, c) := \bigvee_{b \in B} s(a, b) \land t(b, c)
\]

using the sum (disjunction) and multiplication (conjunction) of the semiring $B$. A natural way to generalise this construction is to consider as morphisms matrices over semirings that are closed under set-indexed sums. This brings us to the notion of complete commutative semiring, and the free set-indexed biproduct completion of such a semiring (qua one-object category), which gives rise to a Lafont category [55]. The simplest example of the construction is of course $\text{Rel}$, which is itself the free biproduct completion of the Boolean semiring $\mathbb{B}$.

Recall that a (countably) complete monoid is a pair $(\mathcal{S}, \sum)$ of a set $\mathcal{S}$ with a sum operation $\sum$ on (countably) indexed families of elements of $S$, satisfying

(i) Partition associativity: for every partitioning of the indexing set $I$ into $\{I_j \mid j \in J\}$, $\sum_{i \in I} a_i = \sum_{j \in J} \sum_{i \in I_j} a_i$

(ii) Unary sum: $\sum_{i \in \{j\}} a_i = a_j$.

We write $0$ for the sum of the empty family, which is the neutral element of the sum. Every complete monoid is a commutative monoid in the usual sense, with binary sum $a_1 + a_2 := \sum_{i \in \{1, 2\}} a_i$.

A complete semiring is a tuple $\mathcal{R} = (|\mathcal{R}|, \sum, \cdot, 1)$ where $(|\mathcal{R}|, \cdot, 1)$ is a complete monoid, and $|\mathcal{R}|$, $\cdot$ is a monoid which distributes over $\sum$, i.e., $\sum_{i \in I} (a \cdot b_i) = a \cdot \sum_{i \in I} b_i$ and $\sum_{i \in I} (b_i \cdot a) = (\sum_{i \in I} b_i) \cdot a$. We say that $\mathcal{R}$ is commutative if $(|\mathcal{R}|, \cdot, 1)$ is a commutative monoid.

Free biproduct completion $\mathcal{R}^{\Omega}$ of a complete commutative semiring $\mathcal{R}$: Fix a complete commutative semiring $\mathcal{R} = (|\mathcal{R}|, \sum, \cdot, 1)$. For convenience we use the Kronecker notation $\delta$: given a set $A$ and $a, a' \in A$, define

\[
\delta(a, a') := \begin{cases} 1 & \text{if } a = a' \\ 0 & \text{otherwise} \end{cases}
\]

We define the category $\mathcal{R}^{\Omega}$. The objects of $\mathcal{R}^{\Omega}$ are sets, and the morphisms from $A$ to $B$ are the $(A \times B)$-matrices over $\mathcal{R}$, i.e., elements of $|\mathcal{R}|^{A \times B}$. The identity over $A$ is the diagonal matrix, $\text{id}_A(a, a') := \delta(a, a')$, for all $a, a' \in A$. The
composite of $s \in R^I(A, B)$ and $t \in R^I(B, C)$ is the usual matrix multiplication: for $a \in A$ and $c \in C$

$$
(s \cdot t)(a, c) := \sum_{b \in B} s(a, b) \cdot t(b, c).
$$

The category $R^I$ is $*$-autonomous (actually compact closed). The bifunctor $\otimes : R^I \times R^I \to R^I$ acts like the cartesian product: $A \otimes B = A \times B$, and for $s \in R^I(A, C)$ and $t \in R^I(B, D)$, $s \otimes t : A \otimes B \to C \otimes D$ is defined as

$$(s \otimes t)((a, b), (c, d)) := s(a, c) \cdot t(b, d)$$

The bifunctoriality of tensor follows from the commutativity of $R$. The unit of the tensor is $1 = \{\ast\}$, an arbitrary singleton set. $R^I$ is symmetric monoidal closed: $A \to B = A \times B$; and $ev_{A, B}(((a, b), (a'), b') := \delta((a, b), (a', b'))$, and the exponential transpose is defined as $\lambda(s)(c, (a, b)) := s((c, a), b)$ for $s \in C(C \otimes A, B)$. The object $\bot = 1$ is dualising; it follows that $R^I$ is compact closed.

By construction $R^I$ has products given as disjoint union. Since $R^I$ is compact closed, products are automatically biproducts [43], which we write as $\oplus$. Given a set $I$ of indices, $\bigoplus_{i \in I} A_i := \bigcup_{i \in I} \{i\} \times A_i$, and

$$
\pi_j((i, a), a') := i_j(a, (i, a)) := \delta((i, a), (j, a'))$$

where $\pi_j$ is the projection onto $A_j$, and $i_j$ the injection from $A_j$. The terminal object $\top = \emptyset$.

Since $R^I$ is symmetric monoidal closed and has countable biproducts, it follows from Section III-B that $R^I$ has free linear exponentials given by the Lafont exponentials. Let us spell out the symmetric tensor powers. Given an object $A$ and $n \in \omega$, $eq_{A, n} : M_n(A) \to A^{\otimes n}$ is the equaliser of the $n!$ permutation automorphisms of $A^{\otimes n}$, where $M_n(A)$ is the set of multisets over $A$ of cardinality $n$, and $eq_{A, n}(m, (a_1, \ldots, a_n)) := \delta(m, \{a_1, \ldots, a_n\})$. These equalisers are preserved by tensor product. We construct the Lafont exponentials accordingly:

$$
!A := \bigoplus_{n \in \omega} M_n(A) \cong M_{\text{fin}}(A); \\
\text{der}_A(m, a) := \delta(m, [a]) \\
\text{ctr}_A(m, (m_1, m_2)) := \delta(m, m_1 + m_2) \\
\text{wk}_A(m, *) := \delta(m, [\cdot])
$$

To summarise:

**Proposition 14.** Given a complete commutative semiring $R$, the free biproduct completion $R^I$ is a Lafont category, such that each homset is endowed with the structure of a semimodule over $R$.

**D. Lafont categories from continuous semirings**

Continuous semirings [22, 41] are an important class of complete semirings. A continuous semiring $R = ([R], +, \cdot, 0, 1, \leq)$ is a semiring equipped with a partial order $\leq$ such that $([R], \leq)$ is a directed-complete partial order (CPO) with 0 as the least element, and the operators $+$ and $\cdot$ are continuous. Thanks to directed completeness, we can define $I$-indexed sum over $R$ as

$$
\sum_{r \in I} r := \bigvee_{F \in \text{fin} I} (\sum_{r \in F} r)
$$

for any subset $I \subseteq [R]$. It follows that every continuous semiring has a top element $\infty$, and $p + \infty = \infty$ for all $p \in [R]$.

**Examples:** The following semirings, endowed with their natural ordering, are continuous.

1) Boolean: $B = \{\{t, f\}, \vee, \land, f, t, \leq\}$ where $f < t$.
2) Completed numbers: $N = (\mathbb{N} \cup \{\infty\}, +, \cdot, 0, 1, \leq)$.
3) Completed reals: $\mathbb{P} = (R^+ \cup \{\infty\}, +, \cdot, 0, 1, \leq)$.

A continuous semimodule $(M, +, 0)$ over a continuous semiring $R$ is a semimodule over $R$ with a CPO structure such that 0 is the least element, and addition and scalar multiplication are continuous.

**Proposition 15 ([55]).** Given a continuous commutative semiring $R$, the Kleisli category $R^I$ is a cartesian closed category such that each homset is endowed with the structure of a continuous semimodule over $R$, and composition is continuous.

A Lafont category with countable biproducts has a canonical enrichment over the category of countably complete monoids (see e.g. [51]), but it may not be CPO-enriched. However, Laird showed [51] that by constructing a bifree algebra (i.e. an initial algebra for which the inverse is a terminal coalgebra) for the free exponential, one can construct fixpoints of morphisms of the Kleisli category without assuming any order-theoretic structure. This is in essence an application of the observation (due to Freyd [35], and Simpson and Plotkin [74]) that uniform fixpoint operators exist (and are unique) for any comonad which is an algebraically compact functor [3]. These fixpoints correspond to infinite sums of finitary approximations indexed over nested finite multisets, each representing a unique call-pattern for computation of the fixpoint.

**E. Applications to quantitative program analysis**

By choosing appropriate continuous commutative semirings $R$, the Kleisli category $R^I$ can be used to model interesting quantitative operational properties of higher-order computation [55]. Consider the nondeterministic functional language PCF, which is PCF [70] augmented with nondeterministic branching. The interpretation of PCF-terms in the CPO-enriched cartesian closed category $R^I$, written $[\Gamma \vdash M : A]^R$, is standard. The base type $\text{int}$ is interpreted as the set of natural numbers. The semantics of the PCF-terms is determined by the cartesian closed structure of $R^I$, interpreting the fixpoint operator $Y_A$ by the least fixpoint construction. For the branching term, we set $[\Gamma \vdash M \text{ or } N]^R := [\Gamma \vdash M]^R \cup [\Gamma \vdash N]^R$, where $+$ is the sum operation of the corresponding homset.

**Theorem 16 ([55]).** For all PCF-terms (closed term of type $\text{int}$) $M$ and all $n \geq 0$, $[\Gamma \vdash M]^\mathbb{N}(\ast, n)$ gives the number of reduction sequences from $M$ to the numeral $n$. 

\[^3A\text{ semimodule over a semiring is like a module over a ring except that it is only required to be a commutative monoid (rather than an abelian group).\]
Laird et al [55] showed that, by using the appropriate continuous commutative semiring, the weighted relational model can be used to reason about such quantitative properties of \( \text{PCF}^\text{ct} \) as may- and must-convergence, and compute such quantities as the probability of convergence, and the minimum and maximum number of reduction steps to convergence.

Weighted relations have shown themselves to be a versatile model for the quantitative analysis of computation. In recent work [52], Laird has given a \( R \)-weighted relational model of the \emph{solos calculus} [57], which presents an elegantly economical syntax for describing name mobility in distributed systems. The semantics of a solos term is a matrix over a complete semiring \( R \); it corresponds to the sum in \( R \) of the values associated to the independent reduction paths of the term. The semantics is fully abstract with respect to the sum-of-path evaluation.

Interestingly Laird [53] shows that there is a systematic way to construct accurate quantitative models by transformation from intensional qualitative models (such as games), using the technique of \emph{change of base of an enriched category} [17]. This transformation is induced by a monoidal functor from the category of coherence spaces to the category of modules over a complete semiring. Applying this transformation to the game semantics [2] of Idealized Algol (which bears a natural enrichment over the category of coherence spaces), one obtains a \( R \)-weighted relational model which is fully abstract.

IV. PROFUNCTORS AND GENERALISED SPECIES OF STRUCTURES

This section is about the second generalisation of the relational model, in a 2-categorical direction:

<table>
<thead>
<tr>
<th>Relational Model</th>
<th>2-categorical Generalisation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Category ( \text{Set} )</td>
<td>2-category ( \text{Cat} )</td>
</tr>
<tr>
<td>Category ( \text{Rel} )</td>
<td>Bicategory ( \text{Prof} )</td>
</tr>
<tr>
<td>Comonad ( \mathcal{M}_{\text{fin}} )</td>
<td>Pseudo-comonad ( \mathbb{P} )</td>
</tr>
<tr>
<td>Category ( \mathcal{M}_{\text{Rel}} )</td>
<td>Bicategory ( \text{ESP} )</td>
</tr>
</tbody>
</table>

We introduce the bicategory \( \text{Prof} \) of profunctors, which corresponds to \( \mathcal{R} \); and the cartesian closed bicategory \( \text{ESP} \) of \emph{generalised species of structures}, which corresponds to \( \mathcal{M}_{\text{Rel}} \). We then discuss a recent development [76] of the generalised species of structures as a quantitative model of the nondeterministic \( \lambda \text{Y} \)-calculus.

A. The bicategory of profunctors

The relational model may be generalised in another way, categorically, to a 2-dimensional level. Given small categories \( A \) and \( B \), a \emph{profunctor} (or \emph{distributor}; see e.g. [4,5] and [8, §7.7]) from \( A \) to \( B \) is a functor \( f : A \times B^{\text{op}} \to \text{Set} \), which is written \( f : A \to B \). Profunctors are a categorical generalisation of relations: a relation \( R \subseteq A \times B \) can be seen as a profunctor \( f_R \) between sets (\emph{qua discrete categories}) such that \( f_R(a,b) \) is either a singleton set or the empty set, depending on whether \( (a,b) \in R \). One should view \( f : A \to B \) as a set-valued relation between the categories \( A \) and \( B \). The bicategory \( \text{Prof} \) has small categories as 0-cells; profunctors as 1-cells, and natural transformations between profunctors as 2-cells. The identities are just the hom-functors. Take profunctors \( f : A \to B \) and \( g : B \to C \). A morphism \( \varphi \in B^{\text{op}}(b',b) \) acts on elements in the set \( f(a,b') \) by \( y \mapsto f(a,\varphi)(y) \in f(a,b) \) and in \( g(b,c) \) by \( z \mapsto g(\varphi,c)(z) \in g(b',c) \), which we write \( y[\varphi] \) and \( \{\varphi\}z \), respectively. The composite \( f ; g \) is defined by

\[
(f ; g)(a,c) := \prod_{b \in B} (f(a,b) \times g(b,c)) / \sim
\]

where \( \sim \) is the least equivalence relation containing \( (y,[\varphi])z \sim (y[\varphi],z) \), for every morphism \( \varphi \in B \). Equivalently the composite profunctor \( f ; g : A \to C \) can be described by a coend formula

\[
(f ; g)(a,c) = \int^b f(a,b) \times g(b,c).
\]

The 1-cells of \( \text{Prof} \) are functors \( F : A \times B^{\text{op}} \to \text{Set} \), and so correspond to functors \( \lambda(F) : A \to \hat{B} \), writing \( \hat{B} = [B^{\text{op}}, \text{Set}] \) for the presheaf category. Because presheaf categories are free colimit completions, profunctors from \( A \to \hat{B} \) correspond to colimit-preserving functors between presheaf categories from \( A \) to \( \hat{B} \). The bicategory \( \text{Prof} \) is equivalent to the 2-category \( \text{Coconut} \) whose 0-cells are small categories, whose 1-cells are colimit-preserving functors between the corresponding presheaf categories, and whose 2-cells are the natural transformations between such functors. The bicategory \( \text{Prof} \) has enough structure to model linear logic; it is what may be called a \emph{compact closed bicategory} [16, 48]. With a suitable choice of pseudo-comonad (and there are several such constructions that satisfy the Seely isomorphisms [16, §9]), \( \text{Prof} \) can be made into a (degenerate) Seely model [60] of linear logic in an appropriate bicategorical sense.

B. Generalised species of structures

We give a direct description of the Kleisli bicategory of a pseudo-comonad on the bicategory \( \text{Prof} \) of profunctors. First a notation. Let \( C \) be a small category, we write \( \mathcal{P} \) for the presheaf category. Because presheaf categories are free colimit completions, profunctors from \( A \to \mathcal{P} \) correspond to colimit-preserving functors between the corresponding presheaf categories, and whose 2-cells are the natural transformations between such functors. The bicategory \( \text{Prof} \) has enough structure to model linear logic; it is what may be called a \emph{compact closed bicategory} [16, 48]. With a suitable choice of pseudo-comonad (and there are several such constructions that satisfy the Seely isomorphisms [16, §9]), \( \text{Prof} \) can be made into a (degenerate) Seely model [60] of linear logic in an appropriate bicategorical sense.
where $\theta_1 \cdots \theta_k$ is the concatenation of lists. Given $f : \mathbb{P}A \to B$ and $g : \mathbb{P}B \to C$, their composite $f ; g : \mathbb{P}A \to C$ is defined as the profunctorial composite $f^2 ; g$.

Fiore et al. [33, 34] have shown that the bicategory ESP is cartesian closed, and can be seen as the Kleisli bicategory of a pseudo-comonad $\mathbb{P}$ on the bicategory Prof of profunctors whose action on 0-cells coincides with that of $\mathbb{P}$.

Generalised species of structures are a generalisation of both Girard’s normal functors [40], which are functors $X \to \text{Set}$, where the set $X$ is the denotation of a type, and Joyal’s combinatorial species [46, 47], which are functors $\mathbb{P} \to \text{Set}$, where $\mathbb{P}$ is the category of finite cardinals and bijections. There are obvious connections between between these two types of functors, and Hasegawa [42], amongst others, has investigated the connections. There is however an important difference between the two: the domain of (the power series representation of) a normal functor is any set, by which one can interpret a type; whereas a combinatorial species uses $\mathbb{P}$, which has non-trivial (iso)morphisms. Unifying them, generalised species of structures between small categories have domains with enough variations to interpret types and non-trivial (iso)morphisms.

C. ESP-semantics of nondeterministic $\lambda Y$-calculus

Profunctors are a good semantic basis for quantitative analysis of higher-order computation. Because the bicategory of profunctors is a proof-relevant refinement of the relational model, we can expect the bicategorical framework of generalised species of structures to support a more intensional and precise quantitative analysis. The use of profunctors as a semantic model of computation goes back to the work of Winskel et al. from the 1990s [15, 16, 79]; their motivation was a need of a domain theory for concurrency with a satisfactory account of bisimulation. Also relevant is Hyland’s expansive study [45], generalising domain theory from the relational model to give a range of models based on the bicategory of profunctors. More recently Tsukada and Asada [75] gave a profunctorial reformulation of the HO/N games [44] in which plays are graphs: they constructed a pseudofunctor from the category of HO/N games to the bicategory of profunctors, refining a similar result in [78].

As the bicategory ESP is cartesian closed, it is a model of the $\lambda$-calculus. (In fact, it is a differential $\lambda$-(bi)category, and so a model of the differential/ resource $\lambda$-calculus.) But what kind of a quantitative model is it? What is the quantitative property captured by the ESP-semantics? Targeting the simply-typed nondeterministic $\lambda Y$-calculus, $\lambda_{nd}Y$, these questions have been investigated by Tsukada et al. [76]. For simplicity, simple types are generated from a unique atomic type $o$, and (raw) $\lambda_{nd}Y$-terms are defined by:

$$M, N := x \mid \lambda x^A.M \mid M M \mid Y_A M \mid M \oplus_A M$$

where $Y_A$ is the fixpoint operator of type $(A \to A) \to A$ and $M \oplus_A N$ is the nondeterministic branching where $M$ and $N$ are of type $A$.

Given an interpretation of the atomic type $[o]$ as the category with one object $*$ and one morphism, the interpretation of the function types is determined by the cartesian closed structure of ESP (equivalently, compact closure of Prof via the Girard translation):

$$[!A] := \mathbb{P}[[A]]$$

$$[[A \to B]] := [[A \to B]] = \mathbb{P}[[A]]^{op} \times [[B]].$$

Observe that types are interpreted as groupoids. Similarly, interpreting the nondeterministic branching as the disjoint union, and the fixpoint operator $Y_A^{ESP}$ as the least fixpoint construct, the interpretation of a $\lambda_{nd}Y$-term-in-context as a generalised species of structures, $[[M]^{ESP} : \mathbb{P}[[\Gamma]]^{op} \times [[A]] \to \text{Set}$, is determined by the cartesian closed structure of ESP.

D. Rigid resource calculus and rigid Taylor expansion

Tsukada et al. [76] have given a characterisation of the ESP-semantics $[[M]^{ESP}$ as the rigid Taylor expansion of $M$, denoted $[[M]]_{rTay}$. Intuitively $[[M]]_{rTay}$ is the set of ($\sim$-equivalence classes of) linear approximants of $M$ in the form of rigid resource terms, where the relation $\sim$ coincides with the equivalence relation (1) in the definition of profunctor composition.

The rigid resource calculus is a variant of the resource calculus [10, 30, 69] in which bags of arguments are replaced by lists, written $\langle u_1, \ldots, u_n \rangle$. In the rigid calculus, a permutation of elements in a bag is distinct from but isomorphic to the original bag. Raw terms of the rigid resource calculus are defined by:

\[ t, u := x \mid \lambda \bar{x}.t \mid t \mu \mid t \oplus \bullet \mid \bullet \oplus t \mu := \langle u_1, \ldots, u_n \rangle \]

where $\bullet$ is a place holder for the unused part of branching and $\bar{x}$ is a (possibly empty) sequence of variables. Note that (well-typed) rigid resource terms are linear by construction, i.e., each variable appears exactly once. A rigid resource term is designed to describe a reduction sequence of a $\lambda_{nd}Y$-term it approximates. For example, $t \oplus \bullet$ means that the left-branch should be chosen here; since the right branch is irrelevant in this case, a rigid resource raw term simply ignores it. In contrast to the standard resource calculus, reduction of the rigid resource calculus is deterministic, and this is a significant advantage for the quantitative analysis of $\lambda_{nd}Y$-calculus.

Fix an infinite sequence $z_1, z_2, \ldots, z_n, \ldots$ of distinct variables, and write $\bar{z}$ for a prefix of this sequence. Let $x_1 : B_1, \ldots, x_m : B_m \vdash M : A$ be a $\lambda_{nd}Y$-term in $\eta$-long form. Formally

$$[[M]]_{rTay}(\bar{b}, a) := \{ [[t]] : \bar{z} \mid \bar{t} \mid t \vdash M \text{ and } \bar{z} \mid \bar{t} \vdash a \}$$

for $b_i \in \text{Obj}([B_i])$, and $a \in \text{Obj}([A])$. The judgement $\bar{z} \mid \bar{t} \vdash a$, is just type assignment judgement, viewing objects of the groupoid $[[A]]$ as the rigid nondedempotent intersection types that refine $A$. Here rigid nondedempotent intersection types are just the standard nondedempotent intersection types except that an intersection is not finite multiiset but rather finite sequence of types. The judgement $\bar{z} \mid \bar{t} \not\vdash a$, says that the rigid
resource term \( t \) (with free variables \( \vec{z} \)) approximates the \( \lambda_{nd} Y \)-term \( M \) (with free variables \( \vec{x} \)).

As mentioned before, a key result of Tsukada et al. [76] is the coincidence of the ESP-semantics with the rigid Taylor expansion semantics.

**Theorem 17.** For every term-in-context \( \Gamma \vdash M : A \), we have the following natural isomorphism

\[
\left[ M \right]_{ESP} \cong \left[ M \right]_{Tay} : \mathbb{P}[\Gamma^\op] \times \left[ A \right] \rightarrow \text{Set.}
\]

(2)

Recall that each rigid resource term \( t \) that approximates a \( \lambda_{nd} Y \)-term (in the sense of the judgement \( \vec{z} \triangleleft \vec{x} \vdash t \triangleleft M \)) corresponds to a evaluation sequence of \( M \). In fact, there is a one-to-one correspondence between equivalence classes \([t]_{\sim}\) and evaluation sequences of \( \lambda_{nd} Y \)-terms. Note that this is an *intensional* (or proof-relevant) version of the quantitative result for PCF^\* in Section III-E.

**E. Reasoning about coefficients of the Taylor expansion**

The Taylor expansion of a \( \lambda \)-term [30] is a formal sum of (standard) resource terms with real number coefficients. The properties of its support (i.e. terms with non-zero coefficients) are difficult to reason about. A fundamental property of the Taylor expansion is the **commutation property** of the Taylor expansion. The quantitative import of the commutation property was first established by Ehrhard and Regnier [28, 30] and consequently Pagani et al. [68] proved that the commutation property for \( \lambda_{nd} Y \)-terms. Note that this is an intensional (or proof-relevant) version of the quantitative result for PCF^\* in Section III-E.

**Acknowledgements**

Much of this work was done in collaboration with Takeshi Tsukada, and also with Kazuyuki Asada. The author is grateful to Takeshi Tsukada and Michele Pagani for corrections and comments which greatly improved the manuscript, and to EPSRC for financial support via grant EP/M023974/1.

**References**


