Generalized generalized species of structure and resource modalities

Luc Pellissier
IMERL, Universidad de la República
Julio Herrera y Reissig 565 CP11300.
Montevideo, Uruguay
pellissier@fing.edu.uy

We propose to return to the construction carried in [5], where we claimed that

Simply-typed approximations = intersection types derivations

in the precise sense that we built a categorical equivalence between specific type systems (that encompass all well-known intersection type systems used to characterize normalization such as those presented in [1]) and simply-typed approximations, that we realize as *approximation functors*, that arise from the translation of the language into linear logic. By studying these specific functors, we claim that their main feature is that they map the exponential of linear logic into what can reasonably be called a *resource modality*[1], corresponding either to linear, affine or cartesian intersection types. So, we present the story under the slightly different, and less syntactic, slogan:

Intersection type system = multiplicative linear logic + resource modality

These resource modalities are linked with well-know systems. In particular, generalized species of structure [3] can be seen as a strictification of the Kleisli category of the linear resource modality. The study of the link between these different resource modality can shed a new light on the extensional collapse [2], and paving the way for a study of this collapse for dynamic semantics (such as the Geometry of Interaction and ordered combinatory algebras, used to account for forcing and realizability).

Calculi as operads Calculi, whether representing programming languages or proof languages, can be presented as **Cat**-operads, structures with three levels, reminiscent of Girard's *sous-sols* [4, Section 7.1]: they are characterised by their sets of objects (formulæ), of multi-arrows (1-dimensional arrows, or arrows between objects, proofs or programs) and 2-arrows (2-dimensional arrows, or arrows between arrows, of reductions). Such a formalism is rich enough to represent usual variants of the λ-calculus and also classical systems, such as the $\lambda\mu$-calculus and linear logic's proof-nets.

This language offers moreover a unifying point of view on the ways different calculi are connected together. Indeed, a morphism $\mathscr{C} \to \mathscr{D}$ may be interpreted as either a translation of a calculus $\mathscr{C}$ into a calculus $\mathscr{D}$; a type system for the calculus $\mathscr{D}$, refining the types of $\mathscr{D}$ as types of $\mathscr{C}$, and typing the terms in $\mathscr{D}$ with derivations in $\mathscr{C}$, following [7]; and a semantics (which can be static, or dynamic: interpreting the process of normalization and not just the equality of results) for the calculus $\mathscr{C}$, with $\mathscr{D}$ as an algebra for it.

The program proposed here will necessitate the creation of a categorical toolbox: indeed, the counterparts of classical notions in category theory (such as closedness, comonads,...) do not yet have a counterpart in the more involved setting we propose. We will use these notions liberally in this text.

[1] The name is borrowed from [6]. The relationship between the notions would be worthy of investigation.

Submitted to:
TLLA 2018

Type systems and the Grothendieck construction In order for the interpretation of morphisms of **Cat**-operads as type-systems to be sound, we have to require them to have a fibration-like property[2]. In [5], we showed that the equivalence derived from the Grothendieck construction can be caried in the **Cat**-operadic setting and yield a correspondence between type systems $\mathscr{C} \to \mathscr{D}$ and morphisms of **Cat**-operads $\mathscr{D} \to \mathfrak{Dist}$[3] [9, 5.1].

Such morphisms $\mathscr{D} \to \mathfrak{Dist}$ encompass *approximation functors* that map (if we ignore the details due to the fact that every level is parametrized by the levels below) every term in $\mathscr{D}$ to the set of its approximations and every reduction to the relation that replay the same reduction at the level of approximations. These approximation functors can be obtained by translating $\mathscr{D}$ into linear logic and choosing an approximation policy for the exponential of linear logic.

Resource modalities A striking feature of approximation functors, is that they all have a trivial image on the multiplicative fragment of linear logic, and only differ in their interpretation of the exponential ! (indeed, a multiplicative term is its only approximation). The examples constructed in [5] interpret the exponential by one special "comonad" on $\mathfrak{Dist}$: in the case of linear (non-idempotent) intersection types, by the comonad $\mathbb{B}^{\mathrm{op}}$ of free monoidal symmetric category; in the case of cartesian (idempotent) intersection types, by the comonad $\mathbb{F}^{\mathrm{op}}$ of free cartesian category, and so on.

It is tempting to consider these comonads as *resource modalities* on $\mathfrak{Dist}$. Axiomatizing such resource modalities – which imply to understand which of their properties are useful for them to interpret the exponential ! – and finding in which way approximation functors are generated by their choice of a resource modality is thus, after having devised sound categorical definition, the first step of the program.

Generalized species of structure In [3], the authors define the cartesian closed bicategory of *generalized species of structure* (which are distributors from a category of the form $\mathbb{B}^{\mathrm{op}}A$) and show that it is a model of the λ-calculus. In a sense, $\mathfrak{Dist}$ equiped with $\mathbb{B}^{\mathrm{op}}$ is the linear logic counterpart of this bicategory, but operadic, and more importantly, reduction-aware. This motivates to see a morphism $\mathbb{M}A_1, \ldots, \mathbb{M}A_n \to B$, where $\mathbb{M}$ is a resource modality on $\mathfrak{Dist}$ as a *generalized generalized species of structure*. For some modalities, the space of such species might be a model of linear logic, and for some not, reflecting already well-known results.

Studying the resource modalities from this semantic point of view (as opposed to the type-system point of view we stressed earlier) might shed a light on the relationship between typing and interpreting.

The relational collapse It is well-known that the Scott semantics of linear logic is an idempotent intersection type system. It is also the extensional collapse of the relational model [2], which can be seen as a strict version of the species of structure model. This relational collapse might be lifted to the level of generalized generalized species of structure, and thus studied as a morphism of resource modalities.

This would be a first step towards being able to understand the connection between different dynamical semantics of λ-calculus. Indeed, by adapting the relational collapse to a reduction-aware setting, we might be able to capture semantics such as those arising from the geometry of interaction (that are well-known to be connected to the relational model) and implicative structures [8] (whose connections with filter models – and thus idempotent intersection types – we are currently investigating). This would draw a picture where the semantics of linear logic are organized around the two oppositions *static/dynamic* and *extensional/non-extensional*.

[2] Christened *Niefield fibrations*.

[3] $\mathfrak{Dist}$ is thus a specific **Cat**-operad classifying type systems.

References

[1] Henk Barendregt, Wil Dekkers & Richard Statman (2013): *Lambda Calculus with Types*. Cambridge University Press.

[2] Thomas Ehrhard (2011): *The Scott model of linear logic is the extensional collapse of its relational model*. Theoretical Computer Science.

[3] M Fiore, N Gambino, M Hyland & G Winskel (2008): *The cartesian closed bicategory of generalised species of structures*. Journal of the London Mathematical Society 77(1), pp. 203–220.

[4] Jean-Yves Girard (2006): *Le Point aveugle*. Hermann.

[5] Damiano Mazza, Luc Pellissier & Pierre Vial (2018): *Polyadic approximations, fibrations and intersection types*. PACMPL 2(POPL), pp. 6:1–6:28, doi:10.1145/3158094. Available at `http://doi.acm.org/10.1145/3158094`.

[6] Paul-André Melliès & Nicolas Tabareau (2010): *Resource modalities in tensor logic*. Annals of Pure and Applied Logic.

[7] Paul-André Melliès & Noam Zeilberger (2015): *Functors are Type Refinement Systems*. In: Proceedings of the 42nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2015, Mumbai, India, January 15-17, 2015, ACM Press, New York, New York, USA, pp. 3–16, doi:10.1145/2676726.2676970.

[8] Alexandre Miquel (2018): *Implicative algebras: a new foundation for realizability and forcing*. arXiv.org.

[9] Luc Pellissier (2017): *Reductions and Linear Approximations*. Ph.D. thesis.