Rigid Taylor Expansion and Intersection Type Distributors in the Bang Calculus

Giulio Guerrieri
University of Bath, Bath, United Kingdom
g.guerrieri@bath.ac.uk

Federico Olimpieri
I2M, Aix-Marseille Université, Marseille, France.
federico.olimpieri@univ-amu.fr

Since Girard's introduction of *linear logic* [20], the notion of linearity has played a central role in the Logic-in-Computer-Science community. A program is linear when it uses its inputs only *once* during computation (inputs cannot be copied or deleted); while a non-linear program can call its inputs at will. *Via* the exponential modalities ! and ?, linear logic gives a logical status to the operations of erasing and copying data. Since its very beginning, the study of linear logic contributed to unveil the logical nature of resource consumption and the aforementioned *computational* notion of linearity has been linked to the *analytical* one, according to which the differential of a function f at a point x is the best linear approximation for f near x. However, only through the work of Ehrhard and Regnier [14, 15, 16] a notion of *differentiation* and of Taylor expansion for programs (*i.e.*, of λ-terms), has been introduced. Intuitively, the derivative of a λ-term t is a "resource-sensitive version" of t that uses only once its input along the reduction. The *Taylor expansion* of an ordinary λ-term is then the infinite power series of *linear approximants*, *i.e.*, higher order derivatives, that one can associate with it. The Taylor expansion of λ-terms has been used to give alternative proofs of fundamental results in the theory of pure λ-calculus [1] and to characterize notions of normalization for both call-by-name and call-by-value [28, 32, 6, 29]. Intersection types were introduced by Coppo and Dezani [11, 12] as an extension of simple types by means of the (associative, commutative and idempotent) intersection connective $a \cap b$: a term of type $a \cap b$ can be seen as a program of both type a and type b. This kind of type systems have proven to be very useful to characterize various notion of normalization for λ-terms [24]. If we impose *non-idempotency* to the intersection [19, 7] (i.e. $a \cap a \neq a$), we get a "resource-sensitive" intersection type system, in the sense that the arrow type encodes the *exact* number of times that the term needs its input during computation: intuitively, a term typed $a \cap a \cap b$ can be used twice as a program of type a and once as a program of type b. Non-idempotent intersection types allow *combinatorial* characterization of normalization properties and of the execution time of programs [4, 7, 2] and proof-nets [8, 9]. Also, De Carvalho's non-idempotent intersection type system $\mathcal{R}$ is a syntactic presentation of the categorical semantics of λ-calculus given in the category of sets and relations [7]. The two frameworks of non-idempotent intersection types and linear approximation are deeply connected: the normal form of the Taylor expansion of a λ-term is isomorphic to its relational semantics, modulo a congruence given by substitution of atoms [7].

Inspired by [22, 26, 30, 27], we propose a *categorification* of the these two kinds of semantics. The category of sets and relations is replaced by the bicategory of distributors [3, 5]. Distributor-induced semantics of programming languages were already presented in [10, 17]. In particular, Fiore, Gambino, Hyland and Winskel introduced the bicategory of *generalized species of structure* [17], a very rich framework that generalizes both the relational semantics and Joyal's *combinatorial species* [23, 17, 18, 30]. Recently, Tsukada, Asada and Ong [30, 31] presented the

Submitted to:
Linearity/TLLA 2020

rigid Taylor expansion semantics for an η-expanded fragment of non-deterministic simply-typed λ-calculus with fixed point combinator: the linear approximants are *polyadic terms* [26], where all the combinatorial information of resource terms is *explicitly* coded replacing multisets of resources with lists. Tsukada, Asada and Ong proved that this semantics is naturally isomorphic to the generalized species semantics. Inspired by these works and by Mazza, Pellissier and Vial's general theory of intersection types [26], Olimpieri [27] introduced a general method to define *intersection type distributors*, a categorified version of intersection type disciplines: these intersection types are the *syntactic presentation* of bicategorical semantics for pure λ-calculus given by Kleisli bicategories of distributors for suitable pseudomonads.

In our work, we want to precisely state the correspondence between non-idempotent intersection types and linear approximants in this categorified context of [27]. We introduce (non-idempotent) intersection type distributors in an untyped call-by-push-value setting [21, 13, 25], the *bang calculus*. The call-by-push-value paradigm subsumes call-by-value and call-by-name, from both the operational and denotational semantics standpoints [25, 21]. However, we leave the explicit call-by-name and call-by-value factorizations, in the sense of [21], of our intersection type distributors semantics to future works.

Our categorical approach allows the introduction of a suitable *category of types*, where morphisms between types are a generalization of *subtyping*. Given a type morphism $a' \to a$, the intuition is that the type a' *refines* the type a. We believe that the method of building distributor-induced intersection type denotational semantics presented in [27] can be straightforwardly extended in all generality to this new framework, but we leave the proper presentation of it to future works. We focus on the case where the intersection connective is *non-idempotent*, since we want to establish a relationship between this type system and an appropriate notion of linear approximants. The explicit statement of the correspondence is far from trivial, and it needs an extension of the polyadic calculus. This happens because standard polyadic terms [30, 26] cannot encode all the *qualitative* information produced by the *subtyping* feature of intersection type distributors. Indeed, standard polyadic terms express explicitly all the quantitative information on variable occurrences. To add also the qualitative information about the computational role of a variable during computation, we need a *subtyping-aware* polyadic calculus where a variable is given by a standard variable x and a type morphism $f\colon a' \to a$. Thus we introduce a term calculus for *type morphisms*, and we define the *rigid Taylor expansion* of a bang term T as the set of such extended polyadic terms linearly approximating T. Our main result is the proof of the isomorphism between our intersection type distributors semantics and the extended notion of polyadic approximation.

This new subtyping-aware polyadic calculus allows us to define, up to a natural isomorphism, an *explicit* deterministic reduction relation over type derivations; in other words, non-idempotent intersection type distributors determine a "proof relevant" denotational model, i.e., a semantics whose elements are the type derivations, and not the conclusions of type derivations (as in relational semantics). We conclude by showing that our subtyping-aware rigid Taylor expansion of a bang term is naturally isomorphic to its normal form and commutes with normalization. This result extends to the bang calculus setting the result of [30]. It is interesting to notice that our commutation theorem derives completely by semantic considerations, that is not the case for the standard commutation theorem for Taylor expansion [16, 15].

References

[1] Davide Barbarossa & Giulio Manzonetto (2020): *Taylor Subsumes Scott, Berry, Kahn and Plotkin.* *PACMPL* 4(POPL), pp. 1:1–1:23, doi:10.1145/3371069.

[2] Alexis Bernadet & Stéphane Jean Lengrand (2013): *Non-idempotent intersection types and strong normalisation.* *Logical Methods in Computer Science* Volume 9, Issue 4, doi:10.2168/LMCS-9(4:3)2013.

[3] Francis Borceux (1994): *Handbook of Categorical Algebra.* *Encyclopedia of Mathematics and its Applications* 1, Cambridge University Press, doi:10.1017/CBO9780511525858.

[4] Antonio Bucciarelli, Delia Kesner & Daniel Ventura (2017): *Non-idempotent intersection types for the Lambda-Calculus.* *Logic Journal of the IGPL* 25(4), pp. 431–464, doi:10.1093/jigpal/jzx018.

[5] Jean Bénabou (2000): *Distributors at Work.* Lecture notes of a course given at TU Darmstadt.

[6] Alberto Carraro & Giulio Guerrieri (2014): *A Semantical and Operational Account of Call-by-Value Solvability.* In: *Foundations of Software Science and Computation Structures, FOSSACS 2014*, *Lecture Notes in Computer Science* 8412, Springer, Berlin, Heidelberg, pp. 103–118, doi:10.1007/978-3-642-54830-7_7.

[7] Daniel de Carvalho (2007): *Semantique de la logique lineaire et temps de calcul.* PhD thesis, Aix-Marseille Université.

[8] Daniel de Carvalho, Michele Pagani & Lorenzo Tortora de Falco (2011): *A semantic measure of the execution time in linear logic.* *Theoretical Computer Science* 412(20), pp. 1884 – 1902, doi:https://doi.org/10.1016/j.tcs.2010.12.017.

[9] Daniel de Carvalho & Lorenzo Tortora de Falco (2016): *A semantic account of strong normalization in linear logic.* *Inf. Comput.* 248, pp. 104–129, doi:10.1016/j.ic.2015.12.010.

[10] Gian Luca Cattani & Glynn Winskel (2005): *Profunctors, open maps and bisimulation.* *Mathematical Structures in Computer Science* 15(3), p. 553–614, doi:10.1017/S0960129505004718.

[11] Mario Coppo & Mariangiola Dezani-Ciancaglini (1978): *A new type-assignment for lambda terms.* *Arch. Math. Log.* 19(1), pp. 139–156, doi:10.1007/BF02011875.

[12] Mario Coppo & Mariangiola Dezani-Ciancaglini (1980): *An extension of the basic functionality theory for the λ-calculus.* *Notre Dame Journal of Formal Logic* 21(4), pp. 685–693, doi:10.1305/ndjfl/1093883253.

[13] Thomas Ehrhard & Giulio Guerrieri (2016): *The bang calculus: An untyped lambda-calculus generalizing Call-By-Name and Call-By-Value.* In: *Proceedings of the 18th International Symposium on Principles and Practice of Declarative Programming, PPDP 2016*, Association for Computing Machinery, pp. 174–187, doi:10.1145/2967973.2968608.

[14] Thomas Ehrhard & Laurent Regnier (2003): *The differential lambda-calculus.* *Theor. Comp. Sci.* 309(1-3), doi:10.1016/S0304-3975(03)00392-X.

[15] Thomas Ehrhard & Laurent Regnier (2006): *Böhm Trees, Krivine's Machine and the Taylor Expansion of Lambda-Terms.* In: *Logical Approaches to Computational Barriers, Second Conference on Computability in Europe, CiE 2006*, *Lecture Notes in Computer Science* 3988, Springer, pp. 186–197, doi:10.1007/11780342_20.

[16] Thomas Ehrhard & Laurent Regnier (2008): *Uniformity and the Taylor Expansion of ordinary λ-terms.* *Theoretical Computer Science* 403(2-3), doi:10.1016/j.tcs.2008.06.001.

[17] Marcelo Fiore, Nicola Gambino, Martin Hyland & Glynn Winskel (2008): *The cartesian closed bicategory of generalised species of structures.* *J. of the London Mathematical Society* 77(1), pp. 203–220, doi:10.1112/jlms/jdm096.

[18] Nicola Gambino & André Joyal (2017): *On operads, bimodules and analytic functors.* *Memoirs of the American Mathematical Society* 249(1184), p. 0–0, doi:10.1090/memo/1184.

[19] Philippa Gardner (1994): *Discovering needed reductions using type theory.* In: *Theoretical Aspects of Computer Software, International Conference TACS '94*, *Lecture Notes in Computer Science* 789, Springer, pp. 555–574, doi:10.1007/3-540-57887-0_115.

[20] Jean-Yves Girard (1987): *Linear logic.* *Theoretical Computer Science* 50(1), pp. 1 – 101, doi:https://doi.org/10.1016/0304-3975(87)90045-4.

[21] Giulio Guerrieri & Giulio Manzonetto (2018): *The Bang Calculus and the Two Girard's Translations.* In: *Proceedings Joint International Workshop on Linearity & Trends in Linear Logic and Applications, Linearity-TLLA@FLoC 2018*, *EPTCS* 292, pp. 15–30, doi:10.4204/EPTCS.292.2.

[22] Martin Hyland (2017): *Classical lambda calculus in modern dress.* *Mathematical Structures in Computer Science* 27(5), pp. 762–781, doi:10.1017/S0960129515000377.

[23] André Joyal (1986): *Foncteurs analytiques et espèces de structures.* In: *Combinatoire énumérative*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 126–159.

[24] Jean-Louis Krivine (1993): *Lambda-calculus, types and models.* In: *Ellis Horwood series in computers and their applications.*

[25] Paul Blain Levy (1999): *Call-by-Push-Value: A Subsuming Paradigm.* In: *Typed Lambda Calculi and Applications, 4th International Conference, TLCA'99*, *Lecture Notes in Computer Science* 1581, Springer, p. 228–242, doi:10.1007/3-540-48959-2_17.

[26] Damiano Mazza, Luc Pellissier & Pierre Vial (2018): *Polyadic approximations, fibrations and intersection types.* *Proc. ACM Program. Lang.* 2(POPL), pp. 6:1–6:28, doi:10.1145/3158094.

[27] Federico Olimpieri (2020): *Intersection Type Distributors.* Available at `https://arxiv.org/abs/2002.01287`.

[28] Federico Olimpieri (2020): *Normalization, Taylor expansion and rigid approximation of λ-terms.* Available at `http://arxiv.org/abs/2001.01619`.

[29] Michele Pagani, Christine Tasson & Lionel Vaux (2016): *Strong Normalizability as a Finiteness Structure via the Taylor Expansion of λ-terms.* In: *Foundations of Software Science and Computation Structures - 19th International Conference, FoSSaCS 2016*, *Lecture Notes in Computer Science* 9634, Springer, doi:10.1007/978-3-662-49630-5_24.

[30] Takeshi Tsukada, Kazuyuki Asada & C.-H. Luke Ong (2017): *Generalised Species of Rigid Resource Terms.* In: *Proceedings of the 32rd Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS 2017, doi:10.1109/LICS.2017.8005093.

[31] Takeshi Tsukada, Kazuyuki Asada & C.-H. Luke Ong (2018): *Species, Profunctors and Taylor Expansion Weighted by SMCC: A Unified Framework for Modelling Nondeterministic, Probabilistic and Quantum Programs.* In: *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '18, pp. 889–898, doi:10.1145/3209108.3209157.

[32] Lionel Vaux (2017): *Taylor Expansion, β-Reduction and Normalization.* In: *26th EACSL Annual Conference on Computer Science Logic, CSL 2017*, *LIPIcs* 82, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, pp. 39:1–39:16, doi:10.4230/LIPIcs.CSL.2017.39.