A Theory of Call-by-Value Solvability

Beniamino Accattoli[1] and Giulio Guerrieri[2]

Solvability is a central notion in the semantics of the λ-calculus. First studied by Wadsworth [Wad71, Wad76] and Barendregt [Bar71, Bar74], it characterizes which λ-terms can be seen as producing a result, thus denoting *successful* or *meaningful computations*, and which ones cannot. Instinctively, one would identify results with terms having a normal form, and thus, dually, *unsuccessful* or *meaningless* terms as those not having a normal form. Such an approach has two related drawbacks. Firstly, the representation of partial recursive functions associating *undefinedness* with terms not having a normal form is problematic, as it is not stable by composition. Secondly, the equational theory extending β-conversion $=_\beta$ with the identification of all terms not having a normal form is *inconsistent*, that is, it equates all λ-terms.

The Solvable Theory Both drawbacks disappear if meaningful/meaningless terms are rather identified with solvable/unsolvable terms, where *t is solvable if it exists a head context H sending t to the identity*, that is, such that $H\langle t\rangle \to_\beta^* \mathsf{I}$. Roughly, it means that t can be decomposed by an environment that cannot simply discard t. An important operational characterization is due to Wadsworth [Wad76]: *a term t is solvable if and only if the head reduction of t terminates*. The characterization shows how to refine the naive theory, identifying *meaningful* with *head normalizable* rather than *normalizable*. A compositional representation of partial recursive functions can then be given, and the equational theory extending $=_\beta$ by identifying all unsolvable terms—the *minimal sensible theory* $\mathcal{H}$—is consistent.

Semantics of Call-by-Value Many variants of the λ-calculus have emerged. What is usually referred to as *the* λ-calculus could nowadays be more precisely referred to as the *(strong) call-by-name λ-calculus*. It is the most studied of λ-calculi, and yet it is *never* used in applications. Functional programming languages, in particular, often prefer Plotkin's *call-by-value* λ-calculus [Plo75], where β-redexes can fire only when the argument is a *value* and usually further restrict it to *weak reduction* and to closed terms—what we refer to as *Closed CbV* (λ-calculus).

The denotational semantics of the CbV λ-calculus is much less studied and understood than the CbN one (some notable exceptions are [EHRDR92, PRR99, Ehr12, MPRDR19]). This is not by accident: as first shown by Paolini and Ronchi Della Rocca [PRDR99, Pao01, RP04], there are some inherent complications in trying to adapt semantic notions from CbN to CbV. They stem from the fact that, while Closed CbV is a very elegant setting, denotational semantics have to deal with *open* terms, and Plotkin's operational semantics is not adequate for that because of *premature* normal forms—see Accattoli and Guerrieri for extensive discussions [AG16].

One of the complications is that CbV solvability does not admit an *internal* operational characterization akin to Wadsworth's one for CbN, and thus it is not really manageable.

Two Approaches to Call-by-Value Solvability The literature has explored two opposite approaches towards the difficulty of denotational semantics for the CbV λ-calculus:

1. *Disruptive*: replacing Plotkin's CbV calculus with another, extended CbV calculus as to obtain smoother denotational semantics, and in particular an easier theory of solvability;
2. *Conservative*: considering Plotkin's CbV calculus as untouchable and striving harder to characterize semantical notions (see García-Pérez and Nogueira [GN16]).

One of the achievements of the disruptive approach is an operational characterization of solvability akin to Wadsworth, due to Accattoli and Paolini [AP12]. They introduce a CbV λ-calculus with let expressions, called *value substitution calculus* (shortened to VSC), together with a *solving reduction* that terminates if and only if the term is solvable in the VSC. This is akin to what happens in CbN, where solvable terms are those for which head reduction terminates.

A reconciliation of the disruptive and the conservative approaches is obtained by Guerrieri et al. [GPR17]. On the one hand, they embrace the disruptive approach, as they study Carraro and Guerrieri's *shuffling calculus* [CG14], another extensions of Plotkin's calculus which can be seen as a variant over the VSC and where Accattoli and Paolini's operational characterization of solvability smoothly transfers. On the other hand, they prove that *a λ-term t is solvable in the shuffling calculus if and only if it is solvable in Plotkin's calculus*. Therefore, the disruptive extension becomes a way to study the conservative notion of solvability for Plotkin's calculus.

Open Questions about CbV Solvability These works paved the way for a theory of CbV solvability analogous to the one in CbN. Such a theory however is still lacking. For instance, it is unknown whether CbV unsolvable terms can be consistently equated, *i.e.*, whether the *minimal sensible theory by value* $\mathcal{H}_{\mathcal{V}}$ is consistent. Further delicate points concern the characterization of CbV solvable terms via intersection types. In CbN, a term t is solvable if and only if t is typable with Gardner-de Carvalho's *non-idempotent* intersection types [Gar94, dC07, dC18], also known as *multi types*. Moreover, one can extract quantitative operational information about solvable terms, namely the number of steps of the head strategy, which is a reasonable measure of time complexity for λ-terms (see Accattoli and Dal Lago [AD12]), as well as the size of the head normal form, as first shown by de Carvalho [dC07, dC18].

In CbV, there exist characterizations of solvable terms via intersection and multi types [PRDR99, KMRDR21]. Those type systems, however, are *defective*: contrarily to what claimed in those papers, their systems do not verify subject reduction (for [PRDR99] subject expansion also fails). Carraro and Guerrieri [CG14] characterize CbV solvability using relational semantics, but their characterization is not purely semantic (or type-theoretic) because it also needs the syntactic notion of *CbV Taylor-Ehrhard expansion* [Ehr12]. Additionally, from none of these characterizations it is possible to extract quantitative operational information. They all rely, indeed, on the shuffling calculus, for which it is unclear how to extract (from type derivations) the number of commuting conversion steps, and its time cost model is also unclear [AG16].

Contributions We study all these questions, providing also a *quantitative* analysis of solvability via intersection types. Because of the quantitative aspect, we study solvability *in the VSC* rather than in the shuffling calculus. The VSC is indeed a better fit than the shuffling calculus for quantitative analyses, because its number of β steps *is* a reasonable time cost model and can be extracted from multi type derivations, as shown by Accattoli et al. [ACSC21, AGL21].

In particular, we study CbV solvability via multi types. Our contributions are:

1. *Multi types and CbV solvability*: we characterize CbV solvability using Ehrhard's CbV *multi types* [Ehr12], which are strongly related to linear logic. Namely, we prove that a term is CbV solvable if and only if it is typable with a certain kind of multi types deemed *solvable* and inspired by Paolini and Ronchi Della Rocca [PRDR99];
2. *Bounds from types*: refining our solvable types, we extract the number of steps of solving reduction on a solvable term, together with the size of the solving normal form. This study re-casts de Carvalho's results in CbV, but it also requires new concepts.

While solvability is certainly subtler in CbV than in CbN, our contributions show that, if the presentation of CbV is carefully crafted, then a solid theory of CbV solvability is possible. In fact, we obtain a theory comparable to the one in CbN. This is our main achievement.

References

[ACSC21] Beniamino Accattoli, Andrea Condoluci, and Claudio Sacerdoti Coen. Strong call-by-value is reasonable, implosively. In *LICS*, pages 1–14. IEEE, 2021.

[AD12] Beniamino Accattoli and Ugo Dal Lago. On the Invariance of the Unitary Cost Model for Head Reduction. In *RTA*, pages 22–37, 2012.

[AG16] Beniamino Accattoli and Giulio Guerrieri. Open Call-by-Value. In *Programming Languages and Systems - 14th Asian Symposium, APLAS 2016*, volume 10017 of *Lecture Notes in Computer Science*, pages 206–226. Springer, 2016.

[AGL21] Beniamino Accattoli, Giulio Guerrieri, and Maico Leberle. Semantic bounds and strong call-by-value normalization. *CoRR*, abs/2104.13979, 2021.

[AP12] Beniamino Accattoli and Luca Paolini. Call-by-value solvability, revisited. In *Functional and Logic Programming - 11th International Symposium, FLOPS 2012, Kobe, Japan, May 23-25, 2012. Proceedings*, pages 4–16, 2012.

[Bar71] Hendrik Pieter Barendregt. *Some extensional term models for combinatory logics and l - calculi.* PhD thesis, Univ. Utrecht, 1971.

[Bar74] Hendrik Pieter Barendregt. Solvability in lambda-calculi. *Journal of Symbolic Logic - JSYML*, pages 372–372, 01 1974.

[CG14] Alberto Carraro and Giulio Guerrieri. A semantical and operational account of call-by-value solvability. In *Foundations of Software Science and Computation Structures - 17th International Conference, FOSSACS 2014, Grenoble, France, April 5-13, 2014, Proceedings*, pages 103–118, 2014.

[dC07] Daniel de Carvalho. *Sémantiques de la logique linéaire et temps de calcul.* Thèse de doctorat, Université Aix-Marseille II, 2007.

[dC18] Daniel de Carvalho. Execution time of λ-terms via denotational semantics and intersection types. *Math. Str. in Comput. Sci.*, 28(7):1169–1203, 2018.

[Ehr12] Thomas Ehrhard. Collapsing non-idempotent intersection types. In *CSL*, pages 259–273, 2012.

[EHRDR92] Lavinia Egidi, Furio Honsell, and Simona Ronchi Della Rocca. Operational, denotational and logical descriptions: a case study. *Fundam. Inform.*, 16(1):149–169, 1992.

[Gar94] Philippa Gardner. Discovering needed reductions using type theory. In *TACS '94*, volume 789 of *Lecture Notes in Computer Science*, pages 555–574. Springer, 1994.

[GN16] Álvaro García-Pérez and Pablo Nogueira. No solvable lambda-value term left behind. *Logical Methods in Computer Science*, 12(2), 2016.

[GPR17] Giulio Guerrieri, Luca Paolini, and Simona Ronchi Della Rocca. Standardization and conservativity of a refined call-by-value lambda-calculus. *Logical Methods in Computer Science*, 13(4), 2017.

[KMRDR21] Axel Kerinec, Giulio Manzonetto, and Simona Ronchi Della Rocca. Call-by-value, again! In *FSCD*, volume 195 of *LIPIcs*, pages 7:1–7:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[MPRDR19] Giulio Manzonetto, Michele Pagani, and Simona Ronchi Della Rocca. New semantical insights into call-by-value λ-calculus. *Fundam. Inform.*, 170(1-3):241–265, 2019.

[Pao01] Luca Paolini. Call-by-value separability and computability. In *Theoretical Computer Science, 7th Italian Conference, ICTCS 2001, Torino, Italy, October 4-6, 2001, Proceedings*, pages 74–89, 2001.

[Plo75] Gordon D. Plotkin. Call-by-Name, Call-by-Value and the lambda-Calculus. *Theoretical Computer Science*, 1(2):125–159, 1975.

[PRDR99] Luca Paolini and Simona Ronchi Della Rocca. Call-by-value solvability. *RAIRO Theor. Informatics Appl.*, 33(6):507–534, 1999.

[PRR99] Alberto Pravato, Simona Ronchi Della Rocca, and Luca Roversi. The call-by-value λ-calculus: a semantic investigation. *Math. Str. in Comput. Sci.*, 9(5):617–650, 1999.

[RP04] Simona Ronchi Della Rocca and Luca Paolini. *The Parametric λ-Calculus – A Metamodel for Computation*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2004.

[Wad71] Christopher P. Wadsworth. *Semantics and pragmatics of the lambda-calculus*. PhD Thesis, Oxford, 1971. Chapter 4.

[Wad76] Christopher P. Wadsworth. The Relation Between Computational and Denotational Properties for Scott's D_∞-Models of the Lambda-Calculus. *SIAM J. Comput.*, 5(3):488–521, 1976.