A Standard Theory for the Pure Lambda-Value Calculus

Álvaro Garcia-Pérez[1][2] and Pablo Nogueira[2] *

[1] IMDEA Software Intitute, Madrid, Spain
[2] Universidad Politécnica de Madrid, Spain
agarcia@babel.ls.fi.upm.es, pablo.nogueira@upm.es

1 Abstract

In the classical lambda calculus (i.e., pure untyped λK, see [3]) a term M is solvable iff there exists a head context $\mathbf{C}[\ ] \equiv (\lambda x_1 \ldots x_m.[\ ])\, N_1 \ldots N_n$ such that $\mathbf{C}[M] \to_\beta^* I$ [3]. Solvability characterises those terms that have a definite operational meaning, i.e., the terms that are *operationally relevant*.

In the definition of solvability, the role of the head context $\mathbf{C}[\ ]$ is to close an open M if necessary (i.e., $\lambda x_1 \ldots x_m.M$) and to reveal the operational content of $\lambda x_1 \ldots x_m.M$ by providing arguments $N_1 \ldots N_n$ to it. The I in the right-hand-side of the reduction judgement is just a simplification to keep things succinct. The definition above can be proven equivalent to the earliest—and original—definition in [4], which reads 'M is solvable iff there exists a head context $\mathbf{C}[\ ]$ such that $\mathbf{C}[M] \to_\beta^* \mathsf{NF}$' [4, 5, 11].

In the light of the original definition, the meaning of 'operational relevance' can be better grasped. A program P uses a sub-program S effectively whenever P has a definite result R, and a different result R' is obtained when replacing S by a non-equivalent sub-program S' in P [8]. In the classical lambda calculus, the head contexts belong to the programs that use the term in their hole effectively, M is the sub-program, and the definite results are the normal forms. An operationally relevant term is a term that can be used effectively, i.e., a solvable term. Solvability gives rise to a very successful 'standard theory' [1] according to which all the unsolvables can be consistently equated (i.e., the lambda-theory $\mathcal{H}$ in [3]) and where the concomitant notions of sensibility (i.e., satisfying $\mathcal{H}$) and Böhm trees [3] connect theories and models with good operational behaviour [10, 11, 3].

The simplified definition of solvability above has been duly adapted in [8] to the pure version of the lambda-value calculus (i.e., λ_V in [9] without the primitive constants and δ-rules). A term M is v-solvable iff there exists a head context $\mathbf{C}[\ ] \equiv (\lambda x_1 \ldots x_m.[\ ])N_1 \ldots N_n$ where the N_i are *closed values* such that $\mathbf{C}[M] \to_{\beta_V}^* I$. The notion of v-solvability captures operational relevance of terms, but only in the 'weak' version of the calculus in which the redices in abstractions are never contracted (i.e., what is known as the 'lazy lambda-value calculus' [6, 7]). This lazy lambda-value calculus generalises some of the results in the 'lazy lambda calculus' of [1] to a framework with a by-value calling convention. Both [1] and [6, 7] exhibit good model-theoretical properties, in particular the existence of initial models in any of the conventional categories for domains. However, the lazy lambda-value calculus does not answer in full whether a 'standard theory' exists for the unrestricted λ_V. Unsurprisingly, v-solvability does not capture operational

*Research partially funded by the Spanish 'Ministerio de Economía y Competitividad' through projects DESAFIOS10 TIN2009-14599-C03-00 and STRONGSOFT TIN2012-39391-C04-02, and by 'Comunidad de Madrid' through programme PROMETIDOS P2009/TIC-1465. The first author is supported by Comunidad de Madrid grant CPI/0622/2008 and by IMDEA Software Institute.

relevance in λ_V, nor shows how to equate the meaningless terms in that calculus in a consistent way. In fact, the syntactic characterisation of v-solvability is given in terms of λK's β-reduction, not in terms of β_V-reduction [8].

Inspired by the original definition in [4], we introduce the novel λ_V-solvability which characterises operational relevance in the unrestricted λ_V. A term is λ_V-solvable if there exists a head context $\mathbf{C}[\ ]$ such that $\mathbf{C}[M] \to^*_{\beta_v} \mathsf{NF}_V$, where the NF_V's are the full normal forms of λ_V. We generalise *needed reduction* [2] to λ_V, and introduce *ribcage reduction*, a sub-relation of $\to^*_{\beta_V}$ which is somewhat the λ_V analogous of λK's head reduction. The terms with *ribcage normal form* are operationally relevant in λ_V, and hence having a ribcage normal form constitutes the syntactic definition of λ_V-solvability. Differently from [8], this syntactic definition can be given in terms of β_V-reduction. Our approach is enough to prove the Genericity Lemma and the 'preservation of unsolvables' [11, 3] in λ_V. We also provide a novel characterisation of all the complete strategies in λ_V.

A λ_V-unsolvable has an order n, in the spirit of [1], which informs about the maximum number of trailing lambdas in a term which is β_V-equivalent to the λ_V-unsolvable. The λ_V-solvability gives rise to the λ_V-theory $\mathcal{H}_V$, which equates all the λ_V-unsolvables of the same order. We prove consistency of $\mathcal{H}_V$, thus opening up for the definition of λ_V-Böhm trees. The concomitant notion of ω-sensibility (i.e., satisfying $\mathcal{H}_V$) characterises λ_V-theories and λ_V-models with good operational behaviour. Our contributions help to reinstate the validity of full normal forms in λ_V and to lay the foundations of a 'standard theory' for the pure λ_V-calculus.

References

[1] S. Abramsky. The lazy lambda calculus. In D. A. Turner, editor, *Research Topics in Functional Programming*, pages 65–116. Addison-Welsey, Reading, MA, 1990.

[2] Barendregt, Kennaway, Klop, and Sleep. Needed reduction and spine strategies for the lambda calculus. *Information and Computation*, 75(3):191–231, 1987.

[3] H. Barendregt. *The Lambda Calculus, Its Syntax and Semantics*. North Holland, 1984.

[4] H. P. Barendregt. *Some Extensional Term Models for Combinatory Logics and Lambda Calculi*. Ph.D. thesis, University of Utrecht, Utrecht, NL, 1971.

[5] Henk Barendregt. Solvability in lambda caculi. In *Proceedings of the Orléans Congrès de Logique*, 1972.

[6] L. Egidi, F. Honsell, and S. Ronchi della Rocca. The lazy call–by–value λ–calculus. In Andrzej Tarlecki, editor, *Proceedings of Mathematical Foundations of Computer Science. (MFCS '91)*, volume 520 of *LNCS*, pages 161–169, Berlin, Germany, September 1991. Springer.

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

[8] Luca Paolini and Simona Ronchi Della Rocca. Call-by-value solvability. *ITA*, 33(6):507–534, 1999.

[9] G. Plotkin. Call-by-name, call-by-value and the lambda calculus. *Theoretical Computer Science*, 1:125–159, 1975.

[10] D. Scott. Outline of a mathematical theory of computation. In *Proceedings of the 4th Annual Princeton Conference on Information Sciences and Systems*, 1970.

[11] C. Wadsworth. The relation between computational and denotational properties for Scott's D_∞ models. *Siam J. Comput.*, 5(3):488–521, 1976.