Term collections in λ and ρ-calculi

Germain Faure [1]

LORIA, BP 239 54506 Vandoeuvre-lès-Nancy Cedex France

The ρ-calculus, also called the rewriting calculus, originally emerged from different motivations—and from a different community—than the λ-calculus. It was introduced to make explicit all the ingredients of rewriting such as rule application and result [CK01]. *In fine* the ρ-calculus provides an extension of the λ-calculus with additional concepts originating from rewriting and functional programing, namely, pattern-matching, and a structure construction which provides collections of terms.

There are several aspects of the ρ-calculus that have been studied so far. The dynamics of the computations has been studied [FMS05] by defining interaction nets for the ρ-calculus. We can mention also the study of type systems [BCKL03,Wac04] and its application in a proof theory that handles rich proof-terms in the generalized deduction modulo [Wac05]. On a more practical side, the ρ-calculus has been used to give a semantics both to rewrite based languages [CK01] such as ELAN [BKK+98] and to the atelier FOCAL [Mod,Pre03], an environment dedicated to the development of certified computer algebra libraries (ongoing works). Also, the ρ-calculus has been used to implement efficient decision procedures [SDK+03].

The management of collections of terms is crucial in calculi like the ρ-calculus, in logic programming or in web query languages. Typically, matching constraints that are involved in the calculus may have more than one solution —this is also the case for example in programming language like TOM [Tom], Maude [Mau], ASF+SDF [ASF] or ELAN [Ela]—and thus generates a collection of results.

As previously mentioned, the ρ-calculus extends the syntax and the operational semantics of the λ-calculus by providing matching constraints and collections of terms. For example, let $+$ be a commutative symbol, x, y be variables and a, b constants. In the ρ-calculus, the pattern-matching constraint $x[x + y \ll a + b]$, that is the application of the matching constraint $x + y \ll a + b$ to x, reduces to a collection of terms consisting of the two terms a and b and denoted $a \wr b$. In fact, the two solutions of the pattern matching problem $x + y \ll a + b$, respectively $\{x \leftarrow a, y \leftarrow b\}$ and $\{x \leftarrow b, y \leftarrow a\}$, are both applied to the body of the pattern matching constraint x and then we get the two results a and b. The corresponding evaluation rule is given by:

[1] Email: `germain.faure@loria.fr`

This paper is electronically published in
Electronic Notes in Theoretical Computer Science
URL: `www.elsevier.nl/locate/entcs`

$$M[P \ll N] \quad \rightarrow_{\sigma} \quad M\sigma_1 \wr \ldots \wr M\sigma_n$$

where $\{\sigma_i\}_{i=1}^{n}$ is the set of solutions of the matching problem $P \ll N$.

To handle collections of terms, we distribute the structure operator over the application and the abstraction operator:

$$(M_1 \wr M_2)N \quad \rightarrow_{\delta_{app}} M_1 N \wr M_2 N$$

$$\lambda x\,.\,(M_1 \wr M_2) \rightarrow_{\delta_{abs}} \lambda x\,.\,M_1 \wr \lambda x\,.\,M_2$$

The ρ-calculus consists thus of four evaluation rules: the β rule inherits from the λ-calculus, the σ rule to deal with pattern matching constraints and the δ rules to deal with collections of terms.

Different works on the ρ-calculus propose different approaches to deal with collections of terms. They were originally [CK01,Cir00] represented using sets. In more recent works [CLW03,RC], they are represented via a structure construction whose operational semantics is parametrised by a theory (typically a combination of the axioms of associativity, commutativity and/or idempotence) that the user chooses depending on the way s/he wants to deal with non-determinism in the calculus. For example, the original semantics of 'sets of results' is recovered by considering the associative, commutative and idempotent (ACI) theory on structures.

The generality given by those recent works is broken when the matching constraints involved in the calculus may have more than one solutions [Fau05]. In this case the theory on structures cannot be arbitrary and thus collections of results should be represented using sets. Moreover, two different derivations of a ρ-term may only differ on the strategy used for the application of the δ rules. We propose to identify them by assuming that terms are taken modulo the equivalence relation generated by δ rules. More practically, we consider normalized rewriting [Mar96] by splitting the evaluation rules in two sets: the operational semantics now consists only of the two fundamental rules β and the σ and on the other hand we use the δ rules to consider *canonical sets*, that is terms that are always normalized w.r.t. the δ rules.

This is not only a matter of taste since this has strong impact on the fundamental understanding of the calculus. The computational mechanism of the calculus becomes easier to understand since the δ rules are no longer explicit evaluation steps. This opens in particular new possibilities in a deep study of the calculus such as a Böhm theorem [Kri90] for the ρ-calculus.

The same approach can be applied to other calculi like the λ-calculus with a parallel operator. In fact, the first attempt to a denotational (Scott) semantics of the ρ-calculus proposed in [FM05] enlighten a relation between the ρ-calculus and the $\lambda_{\shortparallel}$-calculus. The $\lambda_{\shortparallel}$-calculus was introduced as a λ-calculus that is expressive for domains of parallel functions [Bou94]. Syntactically, it is an extension of the λ-calculus with a parallel operator that distributes left *w.r.t* the application and with the λ-abstraction (as the structure operator of the ρ-calculus). It has been extended to parallel and non-deterministic λ-calculi like in [DCdP93].

Scott models for the ρ-calculus are surprisingly close to the models of the $\lambda_{\shortparallel}$-calculus (the parallel operator, respectively the structures are adequately represented by the join operator) and this suggests a relationship between the structure operator of the ρ-calculus and the parallel operator of the $\lambda_{\shortparallel}$-calculus.

We introduce a new calculus that extends the syntax and the operational semantics of the λ-calculus to deal with canonical sets of terms. This calculus enjoys the Church-Rooser property and gives a new operational semantics for the $\lambda_{\shortparallel}$-calculus (the work of [Bou94] mainly insists on models while in this work, we propose to look at the $\lambda_{\shortparallel}$-calculus from an operational point of view).

Contributions

We propose a new syntax and operational semantics for the $\lambda_{\shortparallel}$-calculus. We introduce a new approach to deal with collections of results that can be applied both to the $\lambda_{\shortparallel}$-calculus and to the ρ-calculus. We also make clear the relationship between both formalisms. Finally, since the standard techniques of rewriting modulo [Hue80,KK99,Ohl98] cannot be applied to prove the Church-Rosser property, the approach followed here—inspired from [HR03] —may be applied likewise in the abstract study of rewriting modulo an equivalence relation. We finally discuss an implementation of the calculus in TOM [Tom] and its link with canonical abstract syntax trees [Rei06].

Acknowledgments

Alexandre Miquel suggests to the author the reading of [HR03]. This was the starting point of the work. We also thank Lionel Vaux, Horatiu Cirstea and Claude Kirchner for useful interactions and comments on this work.

References

[ASF] ASF+SDF. A component-based language development environment. *http://www.cwi.nl/projects/MetaEnv/*.

[BCKL03] G. Barthe, H. Cirstea, C. Kirchner, and L. Liquori. Pure patterns type systems. In *Principles of Programming Languages - POPL2003, New Orleans, USA*. ACM, January 2003.

[BKK+98] P. Borovanský, C. Kirchner, H. Kirchner, P.-E. Moreau, and C. Ringeissen. An overview of ELAN. In *Proc. of WRLA*, volume 15, `http://www.elsevier.nl/locate/entcs/volume15.html`. ENTCS, September 1998.

[Bou94] G. Boudol. Lambda-calculi for (strict) parallel functions. *Inf. Comput*, 108(1), January 1994.

[Cir00] H. Cirstea. *Calcul de réécriture : fondements et applications*. PhD thesis, Université Henri Poincaré - Nancy I, 2000. October 25.

[CK01] H. Cirstea and C. Kirchner. The rewriting calculus — Part I *and* II. *Logic Journal of the Interest Group in Pure and Applied Logics*, 9(3):427–498, May 2001.

[CLW03] H. Cirstea, L. Liquori, and B. Wack. Rewriting calculus with fixpoints: Untyped and first-order systems. volume 3085. Springer, 2003.

[DCdP93] M. Dezani-Ciancaglini, U. de'Liguoro, and A. Piperno. Filter models for a parallel and non deterministic lambda-calculus. In A. M. Borzyszkowski and S. Sokolowski, editors, *Mathematical Foundations of Computer Science 1993, 18th International Symposium*, volume 711 of *lncs*, pages 403–412, Gdansk, Poland, 30 August– 3 September 1993. Springer.

[Ela] Elan. The ELAN system:. *http://elan.loria.fr/*.

[Fau05] G. Faure. A remark on the study of the rho-calculus in presence of symbols defined modulo an equational theory. Draft notes, LORIA, 2005.

[FM05] G. Faure and A. Miquel. Towards a denotational semantics for the rho-calculus. Draft notes, LORIA, 2005.

[FMS05] M. Fernández, I. Mackie, and F.-R. Sinot. Interaction nets vs. the rho-calculus: Introducing bigraphical nets. In *Proceedings of EXPRESS'05, satellite workshop of Concur, San Francisco, USA, 2005*, Electronic Notes in Computer Science. Elsevier, 2005.

[HR03] T. Herhard and L. Reigner. The differential lambda-calculus. *Theoretical Computer Science*, 309, 2003.

[Hue80] G. Huet. Confluent reductions: Abstract properties and applications to term rewriting systems. *Journal of the ACM*, 27(4):797–821, 1980.

[KK99] C. Kirchner and H. Kirchner. Rewriting, solving, proving. A preliminary version of a book available at www.loria.fr/~ckirchne/rsp.ps.gz, 1999.

[Kri90] J.-L. Krivine. *Lambda-Calcul : Types et Modèles*. Etudes et Recherches en Informatique. Masson, 1990.

[Mar96] C. Marché. Normalized rewriting: An alternative to rewriting modulo a set of equations. *J. Symb. Comput*, 21(3), 1996.

[Mau] Maude. The maude system:. *http://maude.cs.uiuc.edu/*.

[Mod] Mod. Modulogic home page. *http://modulogic.inria.fr*.

[Ohl98] E. Ohlebusch. Church-Rosser theorems for abstract reduction modulo an equivalence relation. In *Proceedings of Rewriting Techniques and Applications (RTA-98)*, volume 1379 of *LNCS*, pages 17–31. Springer, 1998.

[Pre03] V. Prevosto. *Conception et Implantation du langage FoC pour le développement de logiciels certifiés.* Thèse de doctorat, Université Paris 6, September 2003.

[RC] Rewriting-Calculus. The ρ-calculus home page. *http://rho.loria.fr/*.

[Rei06] A. Reilles. Canonical abstract syntax trees. In *Proceedings of the 6th International Workshop on Rewriting Logic and its Applications.* Electronic Notes in Theoretical Computer Science, 2006. to appear.

[SDK+03] A. Stump, A. Deivanayagam, S. Kathol, D. Lingelbach, and D. Schobel. Rogue Decision Procedures. In C. Tinelli and S. Ranise, editors, *1st International Workshop on Pragmatics of Decision Procedures in Automated Reasoning*, 2003.

[Tom] Tom. The Tom langage:. *http://tom.loria.fr/*.

[Wac04] B. Wack. The simply-typed pure pattern type system ensures strong normalization. IFIP-WCC TCS, 2004.

[Wac05] B. Wack. A Curry-Howard-De Bruijn Isomorphism Modulo. *Under submission*, 2005.