1 CoLoR: a Coq Library on Rewriting and termination

Frédéric Blanqui[1], Solange Coupet-Grimal[2], William Delobel[2], Sébastien Hinderer[1], and Adam Koprowski[3]

[1] LORIA†, Campus Scientifique, BP 239, 54506 Vandoeuvre-lès-Nancy, France
[2] Laboratoire d'Informatique Fondamentale de Marseille, UMR 6166, Université de Provence, CMI, 39 rue Joliot-Curie, F-13453, Marseille, France
[3] Eindhoven University of Technology, Department of Computer Science, P.O. Box 513, 5600 MB, Eindhoven, The Netherlands

Abstract. Coq is a tool allowing to certify proofs. This paper describes a Coq library for certifying termination proofs.

1.1 Introduction

Termination is an important and difficult problem. Many criteria have been developed over the last years. They are more and more complex and applied on larger and larger systems. For these tools to be used in the certification of critical systems and proof assistants, their results must be certified.

There are two ways for doing this. First, by certifying the tool itself by proving that every result produced by the tool is correct. It is a very hard and time-consuming task. Moreover, every change in the tool requires to redo some proofs. The second way is to certify what is produced by the tool. This is simpler, does not depend on the way the tool is implemented and, indeed, can be used for certifying the results of other tools. This however requires that the tools provide enough information to easily check their results. In both cases, one needs to certify the termination criteria used by the tool.

Coq is a proof assistant and checker with a very expressive language that is used in the certification of critical systems [3]. CoLoR is a Coq library, freely available on `http://color.loria.fr/`, providing definitions and proofs of termination criteria for rewrite systems [7]. It can therefore be used as a basis for certifying the results of termination tools for rewrite systems.

One can already use it for proving by hand the termination of quite a number of systems by combining the various theorems proved in CoLoR. The next step, which we currently work on, is to define a language expressing combinations of termination criteria as used in the current termination tools, and to automatically generate the corresponding Coq proofs. For instance, a termination tool could say that a rewrite system is terminating by providing a polynomial interpretation. Then, we would try to automatically generate

† UMR 7503 CNRS-INPL-INRIA-Nancy2-UHP

in Coq a proof that this interpretation satisfies the conditions required by the theorem on polynomial interpretations already proved in CoLoR.

In the following, we describe the main ingredients of CoLoR: concrete data structures, terms and termination criteria. See the following table to get an idea of the development size.

CoLoR: 33200 lines of Coq code (including blank lines and comments)

Util	**10800**	**Terms**	**17200**	**Termination**	**5200**
List	2900	Varyadic	400	Conversion	200
Vector	1500	WithArity	2800	Filter	300
Polynom	625	SimpleType	14000	PolyInt	250
Multiset	4400			DP	250
Others	1375			RPO	1200
				HORPO	3000

1.2 Libraries on concrete data structures

Since we want termination criteria to be effectively used, we have to use concrete data structures. Since, for some data structures, there was no such library, we had to develop our own libraries. The main libraries are:

- `List` is a library on polymorphic lists extending the Coq standard library. It contains many functions and theorems about lists. It also contains results on lexicographic orderings and permutations.
- `Vector` is a library on polymorphic vectors extending the Coq standard library. It contains many functions and theorems about vectors. It also contains useful tactics.
- `Multiset` is a library on finite multisets [11]. This library is implemented in a generic way as a functor. A module type `FiniteMultisetCore` defines a basic interface for building multisets (basic operations) and proving properties about multisets (axioms satisfied by the basic operations) over some carrier setoid (set with equivalence). Given a module of type `FiniteMultisetCore`, the functor `FiniteMultiset` provides derived operations and proofs of many properties on multisets. An implementation of `FiniteMultisetCore` is provided by using lists. Other (more efficient) implementations could be given but only the basic operations and properties need to be given. The functor `FiniteMultiset` automatically provides the corresponding theory. Finally, another functor `MultisetOrder` provides a definition of the multiset extension of an ordering and a proof that it preserves well-foundedness.
- `Polynom` is a library on polynomials with multiple indeterminates and integer coefficients [8]. The monomial $x_1^{k_1} \dots x_n^{k_n}$ is represented by the vector of natural numbers $(k_1, \dots, k_n)$. A polynomial $\sum_{i=1}^{p} c_i m_i$, where m_i is a monomial, is represented by the list of pairs (c_i, m_i). A polynomial

can therefore have different representations. The library however provides a function giving the unique coefficient of a monomial into a polynomial. It also provides all the basic operations on polynomials (addition, multiplication, power, composition, evaluation on integers) and theorems on monotony.

1.3 Libraries on terms

CoLoR currently provides three different libraries implementing various kinds of term algebras. Each library provides definitions and theorems about one-hole contexts, substitutions and rewriting. Given a set R of rules, $red(R)$ denotes the smallest rewrite relation containing R, that is, the smallest relation stable by context and substitution.

- `Varyadic` is a library on varyadic terms, that is, terms built from variables and applications of symbols to any number of arguments.
- `WithArity` is a library on algebraic terms built from variables and symbols of fixed arity. An algebraic signature is given by a set S of symbols and an arity function $\alpha : S \rightarrow nat$. The well-formedness of algebraic terms, that is, the fact that symbols are applied to the right number of arguments, is ensured by construction as follows: the arguments of a symbol f are represented by a vector of length $\alpha(f)$. The library provides definitions and theorems on algebras and interpretations. An interpretation is given by a set D and a function from D^n to D for every symbol of arity n. The notion of reduction ordering (well-founded rewrite relation) is defined and the Manna-Ness theorem is proved (if R is included in a reduction ordering then $red(R)$ is terminating). Another interesting contribution is a definition of the (constructor) cap and the aliens of a term. The definition uses in an essential way higher-order features and dependent types. Indeed, to a term t, we associate a triplet (k, f, v) where k is the number of aliens of t; f is a function which, to a vector of terms $(t_1, \ldots, t_k)$, associates the term t with the i-th alien replaced by t_i; and v is the vector of aliens of t. This is used in the dependency pair theorem.
- `SimpleType` is a library on simply typed λ-terms with constants and typing à la Church [11]. De Bruijn indices are used for representation of binders [5]. The formalization includes, among other results, definitions of typed terms over arbitrary many-sorted signatures, a substitution operating on typing judgments, an equivalence relation generalizing the concept of α-convertibility to free variables and an encoding of higher-order algebraic terms with arities by simply typed λ-terms. Many simple properties, like subject reduction or decidability of typing have been formalized and strong normalization for terms with arity can be deduced from the HORPO development where well-foundedness of the union of β-reduction and HORPO relation has been proven.

1.4 Termination criteria

In the following, we present the different termination criteria and theorems that can be used to certify the termination of rewrite systems.

- `Conversion`: In this file, it is proved that a relation on algebraic terms is terminating whenever its encoding in varyadic terms so is. Thus, every theorem for varyadic systems can be applied to algebraic systems.
- `Filter` contains definitions and properties on arguments filterings [1]. Given an algebraic signature $(\mathcal{F},\alpha)$, an arguments filtering is given by, for each symbol f, a boolean vector $\pi(f)$ of length $\alpha(f)$. This provides a new algebraic signature $(\mathcal{F},\alpha')$ where $\alpha'(f)$ is the number of components of $\pi(f)$ that are true, and a transformation $\overline{\pi}$ from $\mathcal{T}(\mathcal{F},\alpha)$ to $\mathcal{T}(\mathcal{F},\alpha')$ which erases in the application of a symbol f to arguments $t_1,\ldots,t_n$ every t_i such that $\pi(f)(i)$ is false. This also provides a transformation on relations: given a relation R on $\mathcal{T}(\mathcal{F},\alpha)$, the filtering of R is the relation R' on $\mathcal{T}(\mathcal{F},\alpha')$ such that $\overline{\pi}(t)R'\overline{\pi}(u)$ if tRu. It is proved that R is terminating whenever R' so is, and that R is a weak reduction ordering whenever R' so is.
- `DP`: In this file, given a list of rules R, we define its dependency pairs $dp(R)$ (the pairs (l,t) such that l is the LHS of a rule $l\rightarrow r\in R$ and t is a subterm of r headed by some defined symbol) and the corresponding chain relation (there is a chain between t and u iff there is t' such that t reduces to t' in an arbitrary number of R-steps at non-top positions, and (t',u) is an instance of a dependency pair) [1]. It is then proved that R is terminating whenever the chain relation is well-founded. To our knowledge, this is the first formal constructive proof of this theorem (we do not consider infinite sequences of reductions). We also prove that, given a reduction pair $(>,\geq)$, if R is included in $\geq$ and $dp(R)$ is included in $>$, then $red(R)$ is terminating.
- `PolyInt` contains definitions and properties about polynomial interpretations for algebraic terms [8,2]. A polynomial interpretation is given by, for each symbol of arity n, a monotonic polynomial on n indeterminates (the monotonicity of a polynomial P of n indeterminates is ensured by requiring that, for all $i\leq n$, the coefficient of x_i in P is positive). A term with n variables is then interpreted by a polynomial $[\![t]\!]$ on n indeterminates. Note that, to make this definition in Coq, we use an auxiliary term structure where the number of variables is bounded. By using the evaluation function of polynomials in the domain D of non-negative integers, we get an interpretation in the usual sense, where a symbol of arity n is interpreted by a function from D^n to D. Next, we prove that the ordering on terms obtained by comparing the interpretations ($t>u$ if $[\![t]\!]>[\![u]\!]$) is a reduction ordering. Therefore, by the Manna-Ness theorem, if R is included in $>$ then $red(R)$ is terminating. And, for proving that $l>r$, it suffices to check that the coefficients of $[\![l]\!]-[\![r]\!]-1$ are non-negative.

- MPO contains a definition and proof of well-foundedness of the multiset path ordering [4,6].
- HORPO contains a definition of Jouannaud and Rubio's higher-order recursive path ordering (HORPO) with multiset status [9,10]. The main property of HORPO, justifying its use for proving termination of higher-order rewrite systems, is its well-foundedness and compatibility with β-reductions. That requires showing that the union of HORPO and β-reduction is strongly normalizing, which is the main result of this development. The proof relies on computability predicate proof method due to Tait and Girard. The proofs of computability properties form the essential part of this development. Other important properties that have been proven include: monotonicity, stability under substitution and decidability of the ordering. HORPO triggered other developments, most notably, the library on simply typed λ-terms, multisets and multiset ordering.

References

1. T. Arts and J. Giesl. Termination of term rewriting using dependency pairs. *Theoretical Computer Science*, 236:133–178, 2000.
2. E. Contejean, C. Marché, A. P. Tomás, and X. Urbain. Mechanically proving termination using polynomial interpretations. *Journal of Automated Reasoning*. To appear.
3. Coq-Development-Team. *The Coq Proof Assistant Reference Manual - Version 8.0*. INRIA Rocquencourt, France, 2004. `http://coq.inria.fr/`.
4. S. Coupet-Grimal and W. Delobel. An effective proof of the well-foundedness of the multiset path ordering. *Applicable Algebra in Engineering Communication and Computing*. To appear.
5. N. de Bruijn. Lambda-calculus notation with nameless dummies: a tool for automatic formula manipulation with application to the Church-Rosser theorem. *Indagationes Mathematicae*, 34(5):381–392, 1972.
6. N. Dershowitz. Orderings for term rewriting systems. *Theoretical Computer Science*, 17:279–301, 1982.
7. N. Dershowitz and J.-P. Jouannaud. Rewrite systems. In *Handbook of Theoretical Computer Science*, vol. B, chap. 6. North-Holland, 1990.
8. S. Hinderer. Certification des preuves de terminaison par interprétations polynomiales. Master's thesis, Université Henri Poincaré, Nancy, France, 2004.
9. J.-P. Jouannaud and A. Rubio. The Higher-Order Recursive Path Ordering. In *Proc. of LICS'99*.
10. A. Koprowski. Certified higher-order recursive path ordering. In *Proc. of RTA'06*, to appear in LNCS.
11. A. Koprowski. Well-foundedness of the higher-order recursive path ordering in Coq. Master's thesis, Free University of Amsterdam, The Netherlands, 2004. Technical report TI-IR-004.