INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Project-Team PROTHEO

Constraints, Mechanized Deduction and Proofs of Software Properties

Lorraine

THEME SYM

Activity Report

2006

Table of contents

1. Team

***PROTHEO** is a research project of **LORIA** (Research Laboratory in Computer Science and Control of Lorraine, UMR 7503), a laboratory shared by **CNRS** (National Center for Scientific Research), **INRIA** (National Institute for Research on Computer Science and Control), **UHP** (University Henri Poincaré Nancy 1), **Nancy2** (University Nancy 2) and **INPL** (National Engineering Institute of Lorraine).*

Christophe Ringeissen and Duc Tran moved to the Cassis team. Their activity is mainly described in the Cassis report.

Head of project
Claude Kirchner [DR INRIA, HdR]

Administrative assistant
Chantal Llorens [CNRS]

Research scientists
Frédéric Blanqui [CR INRIA]
Olivier Bournez [CR INRIA, HdR]
Isabelle Gnaedig [CR INRIA]
Hélène Kirchner [DR CNRS, part time, HdR]
Pierre-Etienne Moreau [CR INRIA]

Faculty members
Horatiu Cirstea [MC Nancy2]

External collaborators
Luigi Liquori [CR INRIA, Sophia-Antipolis]

Technical staff
Laika Moussa [INRIA fixed-term engineer until 14/04/06]
Yoann Toussaint [INRIA fixed-term engineer until 31/08/06]
Eric Deplagne [INRIA fixed-term engineer since 22/06/06]

Postdocs
Clara Bertolissi [MENRT until 31/08/06]
Yves Guiraud [INRIA since 01/09/06]
Benjamin Wack [MENRT until 31/08/06]

Ph. D. students
Oana Andrei [INRIA since 01/10/05]
Émilie Balland [MENRT since 01/10/05]
Paul Brauner [MENRT since 01/10/06]
Guillaume Burel [ENS Lyon until 31/08/06, MENRT since 01/09/06]
Germain Faure [MENRT since 01/10/04]
Florent Garnier [INRIA until 30/11/06, Nancy2 half-time ATER since 01/12/06]
Emmanuel Hainry [MENRT since 01/09/04, with Calligramme]
Clément Houtmann [ENS Cachan since 01/09/06]
Radu Kopetz [INRIA since 01/07/05]
Fabrice Nahon [High school teacher since 01/09/02]
Antoine Reilles [CNRS until 30/09/06, INPL ATER since 01/10/06]
Colin Riba [MENRT since 01/10/04]
Anderson Santana [Brazil since 01/10/04]

Internships
Paul Brauner [ESIAL, 01/02/06-30/06/06]
Jérémie Delaitre [ESIAL, 01/06/06-31/08/06]
Clément Houtmann [ENS Cachan, 15/10/05-01/03/06]
Grégory Klein [ESIAL, 23/01/06-23/05/06]

Xavier Koegler [ENS Paris, 12/06/06-12/08/06]
Loredana Tec [INRIA, 01/04/06-31/07/06]
Julien Vanesson [ESIAL, 23/01/06-23/05/06]
Zhen Wang [Hong Kong, 15/09/05-15/02/06]

2. Overall Objectives

2.1. Overall Objectives

The PROTHEO project aims to design and implement tools for program specification, proof of properties and safe and efficient execution.

We are working on environments for prototyping such tools, on theorem provers specialized in proofs by induction and equational first-order proofs, on proof techniques involving constraints and rewrite rules. The project has three strongly connected research domains:

- Constraint solving,
- Mechanized deduction with rewrite rules and strategies,
- Theorem proving based on deduction modulo.

The team develops and maintains several software packages detailed later in this document. They allow us to test our ideas and results as well as to make them available to the community.

3. Scientific Foundations

3.1. Rewriting and strategies

Keywords: *functional programming*, *rewriting*, *rule based programming*, *strategies*.

Rewriting techniques have been developed since the 1970s and have been applied in particular to the prototyping of formal algebraic specifications and to the automated deduction of properties often related to program verification [86].

Rewriting techniques have been also used for describing inductive theorem provers, for verifying the completeness and coherence proofs for equational or conditional specifications, for defining first-order theorem provers, for solving equations in equational or conditional theories. Rewriting has been also applied to specific domains like, for example, the automatic demonstration of geometric properties or the verification of electronic circuits. This rewriting approach has proved extremely useful for simplifying search spaces, or for including decision procedures in general provers.

A common feature of (the evaluation of) functional languages and of theorem provers (including proof assistants) is the study of strategies. These strategies allow one, for instance, to guide computations and deductions by specifying which rule should be applied to which position in the term, or to restrict the search space by selecting only certain branches. In functional programming, we can also mention lazy evaluation and call-by-need strategy. In theorem proving, it is interesting to clearly separate inference rules and control strategies, since the correctness and completeness proofs are easier to obtain when using such an approach. Moreover, it is necessary to have a sufficiently expressive strategy language in order to express iteration, case reasoning, deterministic and nondeterministic choices. We have been studying strategies from the point of view of their specifications and their properties. We use them to formalize proofs in the demonstration and verification tools we develop.

Last but not least, rewriting is a fundamental paradigm for the description of transformations, either functional or not. Starting from our previous works on rewriting and strategies, we have introduced a new formalism generalizing λ-calculus and rewriting that we called rewriting calculus (ρ-calculus, for short) [5]. The notion of ρ-reduction of the rewriting calculus generalises β-reduction by considering matching on patterns which can be more elaborated than simple variables. We have been studying the expressiveness of this general formalism and the properties of its various instances.

3.2. Constraints

Keywords: *combination problem*, *constraint solving*, *satisfiability*.

The notion of constraint has proved to be of main interest in the modeling of various problems taken from a large variety of domains like mechanics, logic and management of human activities. The properties to satisfy are specified as a set of constraints for which it is important to determine if it admits a solution, or to compute a description of all solutions.

In the context of automated deduction, dealing with symbolic constraints on abstract domains like terms is of the greatest interest. For instance, syntactic unification is solving equational constraints over terms, and it is a fundamental notion for logic programming languages and automated theorem provers. The unification problem extends to the case of equational theories, where function symbols may admit some equational properties like the associativity and commutativity [71]. Other symbolic constraint systems may use predicates distinct from equality, like ordering constraints or membership constraints.

We are interested in the problem of combining symbolic constraint solvers for abstract (term-based) domains. We focus on the matching problem, which is the constraint solving process used when applying rewrite rules. The interest in matching is explained by its crucial role in the ρ-calculus, and more generally in rewrite engines.

3.3. Mechanized deduction

Keywords: *constraints*, *deduction*, *induction*, *paramodulation*, *resolution*, *rewriting*.

Developing methods and tools for verifying software is one of our main goals. To achieve it, we develop techniques and automated deduction systems based on rewriting and constraint solving.

Verifying specifications on recursive data structures often relies on inductive reasoning or equation handling, and uses operator properties like associativity or commutativity.

Rewriting, which enables us to simplify expressions and formulas, is now an essential tool for making the automated proof systems efficient. Moreover, a well founded rewriting relation can be used in a natural way to implement inductive reasoning. So we study termination of rewriting, as well as to guarantee termination of programs, in the scope of our study of rule-based programming languages, and to allow reasoning in automated deduction. A special effort is made for develop specific termination proof tools for rewriting strategies. We now also work on correctness proofs of rule-based computations in the case where key properties like termination are not verified.

Constraints allow us to postpone complex symbolic problem solving,so that they can be solved in an expedient way. They also allow us to increase expressiveness of specification languages and to refine proof strategies.

Dealing with unification or orienting constraints with interpreted operators (like associative-commutative ones) gives the hope of obtaining much simpler automated proofs. Implementing these ideas has indeed allowed W. McCune [58], [81] to solve an open mathematical problem. Combining constraints and rewriting based simplifications induces complex problems, either theoretical as for example strategies completeness, or practical as for instance efficient implementation. We explore these techniques from these two point of views.

4. Application Domains

4.1. Application Domains

Keywords: *compilation*, *modeling*, *program transformation*, *proof of properties*, *prototyping*, *specification*, *verification*.

Our research applies to modeling, prototyping and verification of software components. To model these systems, we use rule based languages with constraints and strategies that allow one to quickly prototype applications. In particular the matching capabilities of such languages offer ease of expressivity for protocol specification and verification. We also intend to apply these techniques to model and check reactive as well as hybrid systems.

Constraint satisfiability, propagation and solving is of course in itself a main domain of application and has led to the creation of the Enginest Software company in 2000. Based on constraint solving, Plansuite, one of Enginest's products, is a global solution for transport and logistics planning. It allows users to anticipate, plan, manage and forecast the use of logistic resources.

The (safe) transformation of XML entities is mainly a rule based process. XSLT is one of the language used for that purpose. The combination of rewrite based transformations, strategies and typing is a natural application domain of the concepts and tools we are developing.

5. Software

5.1. Introduction

In this section, we only describe software that are distributed. Other software are developed within contracts and grants but they are not distributed yet (see Section 7).

5.2. Tom

Keywords: *compilation*, *pattern matching*, *rule based programming*.

Participants: Émilie Balland, Radu Kopetz, Anne-Claire Lonchamp, Pierre-Etienne Moreau, Antoine Reilles, Yoann Toussaint.

Since 2002, we have developed a new system called *Tom* [84], presented in [45]. This system consists of a pattern matching compiler which is particularly well-suited for programming various transformations on trees/terms and XML documents. Its design follows our experiences on the efficient compilation of rule-based systems [8]. The main originality of this system is to be language and data-structure independent. This means that the *Tom* technology can be used in a C, C++ or Java environment. The tool can be seen as a Yacc-like compiler translating patterns into executable pattern matching automata. Similarly to Yacc, when a match is found, the corresponding semantic action (a sequence of instructions written in the chosen underlying language) is triggered and executed. *Tom* supports sophisticated matching theories such as associative matching modulo neutral element (also known as list-matching). This kind of matching theory is particularly well suited to perform list or XML based transformations for example. The main idea consists in encoding a DOM object into a term-based representation (a DOM NodeList becomes an associative list-operator), and then perform matching and subterm retrieving using the *Tom* pattern matching facilities. On the one hand, this approach is not comparable to XSLT. But, on the other side, the expressivity is very high since it is possible to combine powerful pattern matching constructs with the expressive power of Java.

Tom is documented, maintained, and available at http://tom.loria.fr and http://gforge.inria.fr/projects/tom.

5.3. CARIBOO

Keywords: *induction*, *innermost outermost*, *local strategy*, *strategies*, *termination*.

Participants: Isabelle Gnaedig, Hélène Kirchner, Laika Moussa.

Since 2002, we have been developing a new proof tool called *CARIBOO* (for Computing AbstRaction for Induction Based termination prOOfs). Presented first in [66], and distributed since 2004, *CARIBOO* is a termination proof tool for rule-based programming languages, where a program is a rewrite system and query evaluation consists in rewriting a ground expression. It applies to languages such as ASF+SDF, Maude, Cafe-OBJ, or ELAN. *CARIBOO* is dedicated to termination proofs under specific reduction strategies, which becomes of special interest when the computations diverge for standard rewriting. It deals in particular with the two basic strategies. The innermost strategy, specially useful when the rule-based formalism expresses functional programs, and central in their evaluation process, local strategies on operators, allowing to control evaluation strategies in a very precise way. The outermost strategy, useful to avoid evaluations known to be non terminating for the standard strategy, to make strategy computations shorter, and used for interpreters and compilers using call by name.

The proof technique of *CARIBOO* relies on an original generic inductive process, instantiated for the different considered strategies. For proving that a given term terminates, we proceed by explicit induction on the termination property, on ground terms with a noetherian ordering, assuming that any term less than the given term terminates (according to the given strategy). Proving termination on ground terms amounts to proving that the rewriting derivation tree starting from any ground term has only finite branches.

The derivation trees are simulated by proof trees starting from patterns built with a defined symbol and variables. They are developed by alternatively using two main concepts, namely narrowing and abstraction, whose definition depends on the considered rewriting strategy. More precisely, narrowing schematizes all rewriting possibilities of the terms in the derivations. The abstraction process simulates the normalization of subterms in the derivations, according to the strategy, by replacing them by special variables, denoting any of their normal forms. By abstraction, induction elegantly allows to simulate normalization of subterms, without to effectively get the normal forms. The proof procedure terminates with success when finiteness has been inductively established for each proof tree.

The induction ordering is specified in a constructive way: it is not given a priori but the proof process generates ordering and abstraction constraints, that are only required to be satisfiable. These constraints are automatically solved, or delegated to an external constraint solver, or to the user.

This year, we have developed an automatic specification translator, to enable *CARIBOO* to deal with the examples of the TPDB [1] data base. The specification in "trs" format are converted in the *ELAN* "spc" fomat, taken into accout by the prover of *CARIBOO*, itself written in *ELAN*.

CARIBOO is documented, maintained and available at http://cariboo.loria.fr/.

5.4. CoLoR

Keywords: *Coq*, *certification*, *proof*, *rewriting*, *termination*.

Participant: Frédéric Blanqui.

CoLoR is a *Coq* [59] library on rewriting and termination. It is intended to serve as a basis for certifying the output of termination checkers like Cariboo, AProVE, Torpa, TPA, ...It contains libraries on terms (with arity, without arity, simply typed), integer polynomials with multiple variables, finite multisets, and the proof of various termination criteria (polynomial interpretations, dependency pairs, arguments filtering, first and higher-order recursive path ordering). *CoLoR* is distributed under CeCILL license on http://color.loria.fr/.

This year, the *CoLoR* library was extended with the certification of the first-order recursive path ordering [60] and of the higher-order recursive path ordering [78]. It was also adapted to the new version of *Coq*. Finally, the library was presented at the International Workshop on Termination [27].

5.5. Rainbow

Keywords: *Coq*, *certification*, *proof*, *rewriting*, *termination*.

[1] http://www.lri.fr/ marche/tpdb/

Participant: Frédéric Blanqui.

Rainbow is a tool for automatically certifying termination proofs expressed in some termination proof grammar. Termination proofs are translated and checked in *Coq* by using the *CoLoR* library. The termination proof grammar is under development with various participants of the annual international competition on termination. *Rainbow* is distributed under CeCILL license on http://color.loria.fr/.

Adam Koprowski modified his termination prover TPA[2] for generating proofs in the *Rainbow* format. Hence, he could certify the termination of 167 systems over the 864 systems (19.3%) of the Termination Problem Data Base[3].

5.6. ELAN

Keywords: *computation*, *deduction*, *rules*, *specification*, *strategies*.

Participants: Eric Deplagne, Claude Kirchner, Pierre-Etienne Moreau.

The *ELAN* system provides an environment for specifying and prototyping deduction systems in a language based on rewrite rules controlled by strategies. It offers a natural and simple logical framework for the combination of computation and deduction paradigms as it is backed up by the concepts of ρ-calculus and rewriting logic. It supports the design of theorem provers, logic programming languages, constraint solvers and decision procedures and offers a modular framework for studying their combination.

ELAN was developed until 2003. It is still documented, maintained and available at http://elan.loria.fr.

6. New Results

6.1. Rewriting

Keywords: *conditional rewriting*, *rewriting*, *rewriting calculus*, *strategies*, *types*.

We have developed the foundational study of the rewriting calculus, studied the combination of conditional rewriting with lambda-calculus and continued to study probabilistic rewrite systems.

6.1.1. *Explicit ρ-calculus*

Participants: Horatiu Cirstea, Germain Faure, Claude Kirchner.

Following the works on explicit substitutions for λ-calculus, we proposed, studied and exemplified [55] a ρ-calculus that handles explicitly the resolution of the matching constraints and the application of the obtained substitutions. We have also shown that the approach is modular and we have introduced a calculus handling explicitly only the substitution application and another one where the matching constraints are solved at the object level while the resulting substitutions are applied at the meta-level [17]. All these calculi can be extended to arbitrary matching theories.

The explicit substitution application studied initially [55] is not optimal since the possible complexity of term traversals is not taken into account. We have thus composed and improved the previous explicit versions and we introduced a calculus that offers support for the composition of substitutions [17]. We proved the confluence of the calculus and the termination of the explicit constraint handling part.

Moreover, in this approach the matching constraints with no solution can be eliminated earlier in the reduction process leading to a more efficient evaluation. This can be achieved by integrating in the explicit calculus the approach already used for the plain calculus [56].

6.1.2. *Graphs in the ρ-calculus*

Participants: Clara Bertolissi, Horatiu Cirstea, Claude Kirchner.

[2] http://www.win.tue.nl/tpa/
[3] http://www.lri.fr/~marche/tpdb/

Starting from the classical untyped ρ-calculus and in collaboration with Paolo Baldan from the University of Venice, we have proposed an extension of the calculus, called *graph rewriting calculus*, handling structures containing sharing and cycles, rather than simple terms [49].

The classical ρ-calculus is naturally generalized by considering lists of constraints containing unification constraints in addition to the standard matching constraints. This leads to a term-graph representation in an equational style where terms consist of unordered lists of constraints. As for the classical ρ-calculus, the transformations are performed by explicit application of rewrite rules as first class entities.

The evaluation rules are adapted to the new syntax leading to an enhanced expressive power for the calculus. In this new formalism we can represent and manipulate elaborated objects like, for example, regular infinite entities.

Several aspects of the calculus have been investigated so far, like its properties (we have also shown that the calculus is confluent over equivalence classes of terms, under some linearity restrictions on patterns) and its relationship with other existing frameworks [13]. We have shown the conservativity of the graph rewriting calculus versus the two formalisms that inspired its design, that is the classical ρ-calculus and a version of the λ-calculus extended with a `letrec` like construct. Moreover, the possibility of representing structures with cycles and sharing [50] naturally allowed us to propose an encoding of term graphs and a simulation of their reductions in the graph rewriting calculus [13].

6.1.3. *ρ-Calculus and Combinatory Reduction Systems*

Participants: Clara Bertolissi, Horatiu Cirstea, Claude Kirchner.

Since λ-calculus and rewriting have complementary features, their combination has been studied in different contexts. We have already shown that λ-calculus and rewriting are generalized by the ρ-calculus, in the sense that the syntax and the inference rules of the ρ-calculus can be restricted to obtain the other two formalisms.

In the prolongation of our works on the expressive power of the ρ-calculus we have analyzed the relation between ρ-calculus and higher order rewriting. We gave a precise definition of the matching in Combinatory Reduction Systems (*CRS*) [16] and we provided a direct translation of *CRS* into the ρ-calculus [51]. We showed that for every reduction in a *CRS* there is a correspondent reduction in the ρ-calculus [16].

We have also addressed the converse issue: how to simulate rewriting calculus derivations by Combinatory Reduction Systems ones. As a consequence of this result, important properties, like standardisation, are deduced for the rewriting calculus [25].

6.1.4. *Distributive rewriting calculus*

Participants: Horatiu Cirstea, Clément Houtmann, Benjamin Wack.

General term rewriting systems and classical guiding strategies have been encoded in the original rewriting calculus [5] by adding an additional operator that intuitively selects one of the elements from a set of results. We have shown that an equivalent operator can be encoded in the current version of the calculus but the encoding is limited in this case to convergent term rewriting systems [56].

We have shown recently that the previously proposed encoding can be extended to the general case, *i.e.* to arbitrary term rewrite systems [36]. For this, a new evaluation rule that enriches the semantics of the structure operator is added and an evaluation strategy is enforced by imposing a certain discipline on the application of the evaluation rules. This strategy is defined syntactically using an appropriate notion of value and is used in order to recover the confluence of the calculus that is lost in the general case.

6.1.5. *Implementation techniques for the Rewriting Calculus*

Participants: Horatiu Cirstea, Germain Faure.

We used the ρ-calculus as an intermediate language to compile functional languages with pattern-matching features, and adapt the evaluation strategies developed for the ρ-calculus to the specific constraints arising from typed functional programs.

In [65] two alternative encodings of the ρ-calculus in interaction nets [79] are proposed (graph rewrite systems which have been used for the implementation of efficient reduction strategies for the λ-calculus). We showed [35] that a combination of these interaction net encodings provides an implementation for *typed* functional languages with pattern-matching where pattern-matching and 'traditional' β-reduction can proceed in parallel, without additional overheads. The compilation of functional programs in the ρ-calculus, and the subsequent interaction net encoding, uncover a new strategy of evaluation which naturally exploits the implicit parallelism of the different rules of the calculus. This methodology gives thus rise to new, efficient strategies of evaluation for functional languages.

6.1.6. Canonical sets of terms

Participants: Horatiu Cirstea, Germain Faure, Claude Kirchner.

Term collections are fundamental in the context of the rewriting calculus but also in logic programming and 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, Maude, ASF+SDF or ELAN) and thus generates a collection of results.

As a first step in the study of the rewriting calculus with non-unitary matching theories, we studied the lambda-calculus with term collections [38].

In the spirit of the normalized rewriting [80], the proposed approach manages term collections at the meta-level by considering only *canonical sets* that is, sets that are normalized for some rules (the ones previously refereed as "administrative simplifications"). The result is a confluent calculus where the computational mechanism becomes easier to understand since only the β-rule is an explicit evaluation step. While the work of [53] mainly insists on models, in this work we propose to look at the parallel lambda-calculus from an operational point of view.

6.1.7. Matching modulo superdeveloppements

Participants: Germain Faure, Claude Kirchner.

We are interested in the optimization of rewrite based programs and since the ρ-calculus can be used to encode such programs, a promising approach seems to be based on the matching modulo ρ-reduction.

Higher-order matching has been already used [88] for automatic (functional) program transformation in an untyped setting. In such a context the normal forms correspond to an approximation usually called superdevelopments [89]. We have thus considered matching equations built over untyped λ-terms and solved them modulo superdevelopments [37]. We claim that the background theory of superdevelopments provides nice intuitions and simplifications of the different proofs of [88] especially w.r.t. the application to second-order matching.

A next step consists in defining the notions of developments and superdevelopments in the context of the ρ-calculus and adapt the proposed algorithms for this case.

6.1.8. Rewriting and probabilities

Participants: Olivier Bournez, Florent Garnier, Claude Kirchner.

Last year, in a papier published in RTA, we introduced the notion of probabilistic rewrite systems and we gave some conditions entailing termination of those systems within a finite mean number of reduction steps.

Termination was considered under arbitrary unrestricted policies. Policies correspond to strategies for non-probabilistic rewrite systems.

This is often natural or more useful to restrict policies to a subclass. This year, in paper [41], we introduced the notion of positive almost sure termination under strategies, and we provide sufficient criteria to prove termination of a given probabilitic rewrite system under strategies. This was illustrated with several examples.

6.1.9. Polygraphs and functional programming

Participants: Yves Guiraud, Frédéric Blanqui.

Albert Burroni's polygraphs [54] provide a graphical formal framework extending several syntactic frameworks such as term rewriting [67] and propositional proof theory [68].

The main objective is to start the realization of a computer environment, called Cat, for the construction and study of low-dimensional polygraphs, such as the ones encountered in rewriting, functional programming and proof theory. Its main purpose is to bolster both theoretical studies and new applications, which will be, in turn, integrated into Cat.

Among such applications, a use of polygraphs in complexity analysis is currently examined together with Guillaume Bonfante (Calligramme, INRIA Lorraine). The usual setting of functional programming, term rewriting, is not adaptated for the study of the complexity of "divide and conquer"-like algorithms, such as the "fast sort", using polynomial termination orders. But polygraphs provide a more natural description of these programs and allow one to recover the complexity of the "fast sort" using simple polynomial interpretations, whose theory is already known [67].

The same kind of interpretations can also be used to prove termination of term rewriting systems which are not simply terminating, yielding a new possible technique for automatically proving termination. With Frédéric Blanqui, the termination tools of Cat will be adaptated for use on the Termination Problems Data Base to check the power of this new method.

Finally, fundations are also studied with Albert Burroni and François Métayer (Preuves-Programmes-Systèmes, Paris 7), mainly to polygraphically understand binding operators such as the abstraction of the λ-calculus or the quantifiers of first-order logic, possibly leading to a new generation of explicit substitutions calculi. Working sessions with Jean-Louis Loday (Institut de Recherche Mathématique Avancée de Strasbourg, CNRS) and Philippe Malbos (Institut Camille Jordan, Lyon 1) are programmed to continue to develop the transfer of rewriting techniques in universal algebra and homology theory.

6.2. Rule based programming

Keywords: *abstract data-types*, *algebraic specification*, *compilation*, *pattern matching*.

We are studying the design and the implementation of rule based programming languages. We are interested in modularizing our technologies in order to make them available as separate elementary tools.

Also, we continue our study on the application of rule based languages. In particular, we study their applications to XML transformations as well as the analysis and the certification of program transformations.

6.2.1. Quotient types in functional programming

Participant: Frédéric Blanqui.

In the Modulogic project of the ACI Sécurité Informatique, we are developing with Pierre Weis (project Cristal, INRIA Rocquencourt) and Thérèse Hardin (LIP6, Paris), a program generation tool, called Moca, that produces construction functions for quotient types in OCaml. Quotient types are types whose values represent equivalence classes modulo some equational theory. Invariants like "the list is sorted", "there is no element occuring twice", ...can be seen as quotient types modulo common algebraic equational theories like commutativity, idempotence, ...Generating construction functions for ensuring such invariants may be difficult. Consider for instance the combination of associativity (for having right combs) and idempotence for a binary constructor C. Building the representive of the equivalence class of $C(x, y)$ when both x and y are the representatives of their equivalence class, requires to test a number of patterns proportional to the size of y. For instance, $C(x, C(y, C(x, y)))$ must be represented by $C(x, y)$. We are already able to generate construction functions for some combinations of such theories [28]. In addition, we provide an option to generate code that ensures that representatives of quotient types are indeed maximally shared. Providing a truly general tool requires further research. In particular, the class of acceptable rewrite rules and equational theories is still unclear.

6.2.2. Formal Island

Participants: Émilie Balland, Claude Kirchner, Pierre-Etienne Moreau, Antoine Reilles, Anderson Santana.

In [22], we have studied the concept of "Formal Island" that gives a general framework to reason about a language which is embedded into another. A first step to modular formal island is made in [23] and a tiny but nice application to modular HTML proposed in [75]. We have also shown that *Tom* is a formal island and used this result to study the certification of pattern-matching compilation for *Tom*.

Our first work has been to formalize the relations between the algebraic terms and the concrete objects representing them. Since *Tom* is data-structure independent, we needed a formal view of the mapping mechanism used in *Tom*, defining a general framework allowing to anchor algebraic pattern-matching capabilities in existing languages like C, Java or ML. This provides a specific instance of what we call a formal island. Then, using a powerful enough intermediate language, which could be viewed as a subset of $C \cap Java \cap ML$, we formalized the behavior of compiled code and defined the correctness of compiled code with respect to pattern-matching behavior. This allowed us to prove the equivalence of compiled code correctness with a generic first-order proposition whose proof could be achieved via a proof assistant or an automated theorem prover. We worked with Damien Doligiez from the Cristal project on how to connect our tool with the Zenon theorem prover, and how to help the theorem prover proving these goals. We then extended these results to the multi-match situation characteristic of the ML like languages. The approach has been implemented on top of the *Tom* compiler, and used to validate some syntactic matching construction. This prototype implementation will be extended to support full multi-match constructions and to automatize the cooperation with Zenon.

6.2.3. Anti-Pattern Matching

Participants: Claude Kirchner, Radu Kopetz, Pierre-Etienne Moreau.

Pattern matching is a concept widely spread both in computer science community and in everyday life. Whenever we search for something, we build a structured object, a pattern, that specifies the features we are interested in. But we are often in the case where we want to exclude certain characteristics: typically we would like to specify that we search for white cars that are not station wagons.

To this end, we have defined in [76] the notion of anti-patterns and their semantics along with some of their properties. We then extend the classical notion of matching between patterns and ground terms to matching between anti-patterns and ground terms. We provide a rule-based algorithm that finds the solutions to such problems and prove its correctness and completeness. Anti-pattern matching is by nature different from disunification and quite interestingly the anti-pattern matching problem is unitary. Therefore the concept is appropriate to ground a powerful extension to pattern-based programming languages and we show how this is used to extend the expressiveness and usability of the *Tom* language.

6.2.4. Strategic Programming in Java

Participants: Émilie Balland, Pierre-Etienne Moreau, Antoine Reilles.

We have defined a new formalism [42], [12] to ensure that data-structures are maintained in canonical form using invariants. This integrates the "private type" notion of OCaml into Java. The goal of this approach is to obtain safe and efficient programming interfaces for abstract syntax trees.

As imagined several years ago, a sophisticated strategy language is needed to control the application of rules and describe complex exploration strategies. Inspired by Stratego, we have defined a strategy language for *Tom*. Starting from elementary strategies such as identity, failure, sequence, congruence, and recursion. More elaborated strategies such repeat, top-down or innermost can be defined. Recently we have defined and implemented a strategy library where the strategy expressions can be parameterized, matched, dynamically rewritten, and explored by strategies itself. Written in Java, in some sense this brings functions to the object level. The presented strategy language has been applied to the development of verification tools [57], [73], [39].

The strategy language has also been used to specify an optimizer for *Tom*. We have introduced a program transformation phase, presented in [24], which optimizes the intermediate program in order to improve its efficiency. As a result, *Tom* generates an efficient implementation of pattern matching, comparable to very optimized implementations such as those in *ELAN* or ASF+SDF.

6.2.5. Rewriting-Based Access Control Policies

Participants: Anderson Santana, Claude Kirchner, Hélène Kirchner.

Recently we proposed a formalization of access control policies based on term rewriting [43]. The state of the system to which policies are enforced is represented as an algebraic term, what allows to model many aspects of the policy environment. Policies are sets of rewrite rules, whose evaluation produces deterministic authorization decisions. We present the relation between properties of term rewriting systems and those important for access control, *e.g.* consistency, as well as the impact of composing policies to these properties.

6.2.6. Rule-Based Modeling for Biochemical Applications

Participants: Oana Andrei, Hélène Kirchner.

Transformations in chemical systems are naturally expressed by rules on some classes of graphs. The work on rule-based programming using *ELAN* for the automated generation of chemical reactions involved in combustion reaction mechanisms, presented in [70], led us to explore how this application can benefit from the formal island approach and from the *Tom* environment [21].

The rule-based approach on graph structures has also been studied in the domains of biological systems and protein interactions. This motivated us in starting to develop a rewriting calculus based on ρ-calculus for modelling the interactions in biochemical systems using a graph-like representation for the biochemical species. The control of biochemical reaction applications can then be expressed by means of strategies as it was already done for the control of chemical reactions.

6.3. Mechanized deduction

Keywords: *completion procedures*, *constrained theories*, *decision procedures*, *deduction modulo*, *equational proofs*, *induction proofs*, *termination*.

We continued to develop both the inductive and size-based approaches to termination of rule-based programs. We also began to develop a *Coq* library for certifying termination proofs. Finally, we developed the use of "good proofs" in automated deduction.

6.3.1. Inductive strong termination proofs

Participants: Isabelle Gnaedig, Hélène Kirchner.

In previous works, we have proposed a method for specifically proving, by explicit induction, termination of rewriting under strategies. The proof principle consists, for a given term rewriting system, in establishing on the ground term algebra that every term terminates i.e. has only finite derivations, assuming that any smaller term terminates. For that, we developed proof trees representing the possible derivations from any term using the term rewriting system. Two mechanisms are used, namely abstraction, introducing variables that represent ground terms in normal form and schematize normalization of subterms, and narrowing, schematizing rewriting in different ways according to the ground instances it represents. These two mechanisms deal with constraints used to control the growth of the proof trees.

After applying this methodology for proving termination of rewriting with the innermost, the outermost and the local strategies, we worked on a factorization of these three procedures, and obtained a generic algorithm with appropriate instances to the different strategies.

This year, we completed this work in studying the ordering constraints generated by our inductive proof principle, in order to automate the proof algorithms for the different strategies. For the three rewriting strategies, we have analyzed the behavior of our approach on the examples of the TPDB termination data base [4]. Thanks to the power of induction, for most examples, the subterm ordering or a lexicographic path ordering are sufficient. Else, a polynomial ordering can verify the constraints. We identified and analyzed other failure cases for ordering constraint solving and proposed a few solutions. This work has been submitted to a journal [19].

[4] http://www.lri.fr/~marche/tpdb/

6.3.2. Sufficient completeness and termination

Participants: Isabelle Gnaedig, Hélène Kirchner.

By its generality, the principle of the inductive method, we have proposed for termination, is promising to handle other properties. So we have studied the possibility to prove sufficient completeness with our approach.

This led us to generalize the well-known result linking sufficient completeness and ground reducibility, in establishing that only weak termination suffices for the second property to imply the first. Moreover, sufficient completeness usually applies to a set of equations, and then extends to a rule-based program if rules are confluent and terminating. Unfortunately, often, programs do not fulfill the last two requirements. To avoid them, we introduced the notion of C-reducibility, expressing that every data evaluates into a constructor form i.e. a completely evaluated expression, for at least one of its possible evaluations. We proposed an algorithm, based on our induction principle, using narrowing and abstraction, to establish C-reducibility, even if the program does not terminate. It is constructive in the sense that a computation branch leading to a constructor form can be deduced from the generated proof trees. As the most important thing for a program is to give an evaluated form for every data, our result relativizes the importance of termination in the context of programming [40].

6.3.3. Higher-order conditional rewriting

Participants: Frédéric Blanqui, Claude Kirchner, Colin Riba.

We extended to conditional rewriting the work of Dougherty [63] and Müller [85] on the combination of λ-calculus and algebraic rewriting. We considered the following two cases: when β-reduction is used or not in the evaluation of conditions. Incidentally, we improved Dougherty's result by requiring weak normalization of β-reduction (*i.e.* existence of a terminating β-reduction) instead of strong normalization (*i.e.* termination of every β-reduction). Finally, we provided a syntactic condition for ensuring orthogonality, hence confluence, in case of conditions with higher-order terms. This is the first study on the confluence of λ-calculus with higher-order conditional rewriting [30]. A journal version has been submitted.

6.3.4. Size-based termination

Participants: Frédéric Blanqui, Colin Riba.

In previous years, we generalized to rewriting and dependent types the so-called type or size based termination criteria for ML-like function definitions. They are based on the way inductive types are usually constructed: as fixpoints of monotone operators on the set of terms. The number of steps necessary for reaching a term in such a fixpoint gives a simple and natural measure, the size, which is finer than the syntactic structure. By enriching the syntax of types with size annotations, one can express that a term has a size (strictly) smaller than another term.

In [31], we extended our previous work by allowing existential and universal quantifications over size variables and contraints over size variables in Presburger arithmetic. This allows one to specify very precisely how the size of the output of a function depends on the sizes of its inputs. For instance, one can automatically check that various sorting algorithms preserve the size of their input. We also showed how this can be used for proving the termination of conditional systems. This provides the first termination criterion for higher-order conditional rewriting.

6.3.5. Termination of higher-order rewrite systems

Participant: Frédéric Blanqui.

In the first-order case, the termination of a rewrite system R is equivalent to the top termination of its dependency pairs modulo R [48]. In [26], we extended this result to the simply typed λ-calculus for some class of rewrite systems by using Tait's technique of computability predicates.

In [44], we studied the relations between the higher-order recursive path ordering (HORPO) [72] and termination criteria based on the computability closure [52]. We proved that, in the first-order case, the fixpoint of the computability closure is exactly RPO. In the higher-order case, we get an ordering that strictly contains HORPO. However, this ordering is difficult to use in practice. In [29], we proposed a new definition of HORPO inspired from the previous work. But this definition remains to be precised and checked in details.

6.3.6. Abstract completion

Participants: Guillaume Burel, Claude Kirchner.

Solving goals, like deciding word problems or resolving constraints, is much easier in some theory presentations than in others. What have been called "completion processes", particularly in the study of equational logic, involves finding appropriate presentations of a given theory to solve easily a given class of problems.

We have designed a general proof-theoretic setting within which completion-like processes can be modeled and studied. This framework (the abstract canonical systems and inference) centers around well-founded orderings of proofs. It allows for abstract definitions and very general characterizations of saturation processes and redundancy criteria [62][18].

In [33], we applied this framework to the standard completion [77], proving that it can indeed be used to cover concrete procedures of completion. To prove this fact, we had to define a interesting translation from proof terms like in the rewriting logic [82] to proof by replacement of equal by equal.

In [34], using the abstract inference, we also designed a new completion procedure which permits to recover the cut-elimination property in deduction modulo [64]. Indeed, a small counter-example proves that, even for convergent rewrite systems which can also rewrite propositions, the cut-elimination property, i.e. the fact that every sequent that is provable can be proved without cut, is no longer true [69]. We first proved that this property is in fact undecidable. Then, we proved that deduction modulo can be seen as an instance of abstract canonical system, in a way such that the critical proofs corresponds to proofs with only one cut as first inference rule, directly followed by rewritings. Therefore, adding the conclusion of such proofs permits to recover the cut-elimination property.

6.3.7. Super deduction

Participants: Horatiu Cirstea, Paul Brauner, Clément Houtman, Claude Kirchner, Benjamin Wack.

Following the seminal work of Benjamin Wack on extended natural deduction [87], we have extended the approach from natural deduction to the sequent calculus [46] and studied the consistency of the super deduction systems [47]. An implementation of these concepts in currently designed and writen in *Tom*. The first results have been presented in particular at the third workshop on the rewriting calculus is London.

6.3.8. Induction and rewriting

Participants: Claude Kirchner, Hélène Kirchner, Fabrice Nahon.

Following our work with Eric Deplagne on a proof theoretic framework to perform rewrite based inductive reasonning [61], we have developed proof search inference systems based on narrowing. This approach does not need to compute induction variables nor induction schema which are automatically infered by the narrowing process. Soundness and completeness of the system have been proved and the system examplified on several cases [74]. This has been extended this year to rewriting modulo equational theories like associative or commutative theories.

6.4. Computability and complexity

Keywords: *analog computations*, *complexity*, *computability*, *implicit complexity*.

We focus on computational models working over reals with a discrete and continuous time. This year we mainly focused on continuous time models. Concerning continuous time models, we have provided machine independent characterizations of computable and elementary computable functions over reals. We have also discussed the computational power of several models.

6.4.1. Analog computations

Participants: Olivier Bournez, Daniel Graça, Emmanuel Hainry.

There are many ways to model analog computers. Unifying those models is therefore a crucial matter as opposed to the discrete case, as there is no property stating that those models are equivalent. It is possible to distinguish two groups in these models: on one side, continuous time models; on the other side, discrete time models working on continuous structures as a model derived from Turing machines. The first group contains in particular some sets of functions defined by Moore in [83]. The main representative of the second group are the real computable functions and a subclass of this set: the set of elementary computable functions.

There are few comparisons between classes of functions from the first group and from the second group, and in particular, there were almost no result of equality.

It is known that Euler's Gamma function is computable according to computable analysis, while it cannot be generated by Shannon's General Purpose Analog Computer (GPAC). This example has often been used in order to argue that the GPAC is less powerful than digital computations.

However, as we demonstrated in [32], when computability with GPACs is considered in the framework of recursive analysis, we obtain two equivalent models of analog computation. Since GPACs are equivalent to systems of polynomial differential equations then we show that all real computable functions can be defined by such models. We also submitted an extended version of this conference paper to journal of complexity [15].

In [14], in collaboration with Manuel Campagnolo in Lisbon, we wrote a survey on continuous time models of computations. In this paper, we provide an overview of computation theories of continuous time computation. These theories allow us to understand both the hardness of questions related to continuous time dynamical systems and the computational power of continuous time analog models. We survey the existing models, summarizing results, and point to relevant references in the literature.

Some of the results above are part of the PhD thesis of Emmanuel Hainry. Emmanuel Hainry defended his PhD thesis in December [11], whose abstract follows:

"Computation on the real numbers can be modelised in several different ways. There indeed exist a lot of different computation models on the reals. However, there exist few results for comparing those models, and most of these results are incomparability results. The case of computation over the real numbers hence is quite different from that of computation over integer numbers where Church-Turing's thesis claims that all reasonable models compute exactly the same functions.

The results presented in this document are twofold. One, we show that recursively computable functions (in the sense of computable analysis) can be shown equivalent to some adequately defined subclass of $\mathbb{R}$-recursive functions, and also to gpac-computable functions with gpac-computable roughly meaning computable through a converging gpac. Hence we get a machine independent characterization of recursively computable functions, and a bridge between type 2-machines and gpac. Two, more than an analog characterization of recursively enumerable functions, we show that the limit operator we defined can be used to provide an analog characterization of elementarily computable functions and $\mathbb{E}_n$-computable functions for $n \geq 3$, where $\mathbb{E}_n$ represents the levels of Grzegorczyk's hierarchy.

Those results can be seen as a first step toward a unification of computable functions over the reals."

7. Contracts and Grants with Industry

7.1. Averroes

Participants: Frédéric Blanqui, Olivier Bournez, Horatiu Cirstea, Claude Kirchner.

The main goal of the RNTL Averroes project (Analysis and VERification for the Reliability Of Embedded Systems) is to contribute to the development of formal methods able to verify multi-form (quantitative) functional and non-functional properties, that appear in industrial problematics. The participants of this project are France Télécom R&D, CRIL Technology Systèmes Avancés, the LaBRI in Bordeaux, the LORIA in Nancy, the PCRI in Saclay and the LSV in Cachan.

The project relies on results obtained in previous National Network for Research on Telecommunication (RNRT) project Calife. It aims at consolidating some of them (e.g. implementation of theoretical results in proof tools) or at generalizing the considered framework in order to solve problems that appeared in practice. For instance, the consideration of stochastic systems, or of properties dealing with consumption of resources. A description of the project can be found at http://www-verimag.imag.fr/AVERROES/.

We are particularly involved in Lot 2 ("applications'1'), concerning applications, in Lot 3 ("tests and animations"), concerning the graphical animation of execution traces in the platform developed by the project, in Lot 4 ("model technologies"), concerning modeling technologies for probabilistic and stochastic systems, and for guaranteeing complexity bounds on consumption of resources, and in Lot 5 ("verification technologies"), about adding rewriting to *Coq*, and about the mechanization of deduction.

7.2. Manifico

Participants: Horatiu Cirstea, Claude Kirchner, Radu Kopetz, Anne-Claire Lonchamp, Pierre-Etienne Moreau.

The main goal of the Manifico project is to improve the expressivity, the efficiency and the modularity of rule based systems. In particular, we are interested in studying how the use of constraints can improve the expressivity of rule based languages. The Protheo and the Contraintes INRIA teams are involved in this project. **ILOG** is the industrial partner.

By studying and understanding the relationship between pattern matching, RETE algorithm and constraint solving, one of our goals is to develop some new compilation algorithms which can combine the best of these three technologies.

8. Other Grants and Activities

8.1. Glossary

AICA	Associazione Italiana per l'Inforatica ed il Calcolo Automatico
ANR	National Agency for Research
ARA	ANR Fundamental Research Action
ARASSIA	ARA on Security, embedded Systems and Ambient Intelligence
ACI	FNS Concerted and Incentive Action
ACISI	ACI on Computer and Software Security
CISSI	Comité interministériel pour la sécurité des systèmes d'information
Compulog	ESPRIT network on Computational Logic
CSD	Conseil Scientifique de la Défense
CPER	Planning Contract between the Government and the Region
ERCIM	European Research Consortium for Informatics and Mathematics
ESPRIT	Information Technologies Program of the European Union
FNS	National Fund for Science
GDR	CNRS Research Group

GDR ALP	GDR on Algorithmics, Languages and Programming
PAI	Integrated Action Programme
PRST	CPER Pole of Scientific and Technological Research
PRST-IL	PRST devoted to Software Intelligence
QSL	PRST-IL project on Quality and Safety of Software
RNTL	National Network on software Technology

8.2. Regional initiatives

In the context of the CPER 2000-2006, we are involved in the regional Research and Technology Project on Software Intelligence (PRST-IL) coordinated by Hélène Kirchner. PROTHEO is involved in the PRST-IL theme on Quality and Safety of Software with the TOUNDRA project on toolboxes for program proofs in cooperation with Calligramme. We also obtained a financial support from the region for the development of *CoLoR*.

8.2.1. Toundra

Participants: Isabelle Gnaedig, Hélène Kirchner.

The main goal of the Toundra action "Toolboxes for Program Proofs" is to develop integrated toolboxes for verifying and proving program properties of rule-based languages. Our aim is to provide expertise-encapsulated environments allowing non specialists to certify programs.

To achieve our goal, we first have to develop property proof algorithms for rewriting, that are specific to the context of programming. In fact, there already exists a large amount of proof techniques for rewriting properties but, most of the time, they are not adapted to the context of programming. Finer tools could allow working with the initial semantics instead of the free one, and with specific strategies instead of the general rewriting relation.

Second, we have to study how these tools can cooperate, in the most efficient way, to cover the largest possible applicability domains and allow non-expert users.

To this end, we develop termination proof algorithms for specific strategies, like innermost and outermost strategies, local strategies on operators, or *ELAN*-like strategies. We also study applicability and complexity arguments on these tools, as well as their possible relations in order to develop an expertise kernel for managing the different proof tools.

The project has been terminated and evaluated this year.

8.3. National initiatives

We participate in the following GDR ALP projects: COLLAB (collaboration of solvers) and "Logic, Algebra and Computation" (mixing algebraic and logical systems).

8.3.1. Infer

Participants: Guillaume Burel, Claude Kirchner.

This new project is a grouping of three teams through their common interest for a new approach to proof theory, called "deep inference", that has been developed during the last five years by a group of researchers centered around Alessio Guglielmi. We aim at refining its potential and at applying it to problems related to the foundations of logic and to more practical questions in the algorithmics of deductive systems.

Among the list of theoretical problems there is the fundamental need for a theory of correct identification of proofs, and its corollary, the development of a really general and flexible approach to proof nets. A closely related problem is the extension of the Curry-Howard isomorphism to these new representations. Among the list of more practical problems we will tackle questions of strategy and complexity in proof search, in particular for higher-order systems. These questions are intimately related to how proofs themselves are formulated in these systems, and the obvious relationship betwen deep inference and well established techniques—like deduction modulo and unification for quantifiers—are subjects that we intend to deepen, given their common grouding in rewriting theory. The project also intend to explore the formuation and use of more “exotic” logical systems, for example noncommutative ones, that have interesting applications, as in linguistics and quantum computing. This is a natural continuation of some of the first results that were obtained in deep inference.

8.3.2. Modulogic

Participants: Frédéric Blanqui, Horatiu Cirstea, Isabelle Gnaedig, Pierre-Etienne Moreau, Laika Moussa, Antoine Reilles.

Modulogic is an ACISI project started in September 2003. Its main goal is to build an integrated toolbox for asserted software. This toolbox allows writing modules built on declaration, definitions, statements and proofs. Declarations can be refined into definitions and statements into proofs, by progressively migrating from the specification to the implementation, with inheriting and redefining mechanisms, and by instantiating parameters.

The INRIA Protheo team and the Mirho action, as well as the FOC group (SPI-LIP6, Paris 6, CEDRIC, CNAM) and the Cristal project are also involved.

The integrated toolbox we aim to build, will offer an appropriate interface for interactions with compilers of programming languages, with proof verifying systems, and with proof search systems, allowing to automatically refine statements. Definitions and some parts of the proofs will be let to the user. This toolbox will then be used for interacting with existing tools and languages (like Caml, *Coq* ...). The originality of our approach lies on a formal integration of these tools.

Contributions we would like to develop are the following: to conceive and realize the toolbox, to develop the component “proof research tool”. They will be developed in order to be applied to safety strategies. In fact, we think that formal certification of safety properties can only be thought in a global way, and such a toolbox should help us to do it.

Building the toolbox can be seen as the continuation of the development of the systems Foc and *ELAN*. The ongoing work on the semantics of the object oriented languages and on the ρ-calculus will constitute the basis of the semantics for Modulogic. Building efficient proof research tools will be based on our work on the deduction modulo and on our expertise in the domain of rewriting. Adapting the toolbox to the safety strategies will be made possible thanks to our expertise for specifying safety properties in *Coq*, in particular the model of Bell and Lapadula.

8.3.3. Inval

Participant: Yves Guiraud.

The ANR project "Invariants algébriques des systèmes informatiques", coordinated by Éric Goubault (CEA Saclay), has started in January 2006 and federates researchers in computer science from CEA Saclay, Paris 7, INRIA Futurs and INRIA Lorraine on one side, and researchers in mathematics from CNRS-Strasbourg, Lyon 1, Caen and Aix-Marseille 2. Its main objective is to continue the growing contacts and transfers of ideas between both communities, mainly after the ACI "Géométrie du calcul".

8.4. International networks and working groups

We are involved in the COMPULOG network which consists of a lot of groups working on logic programming. We participate to the working groups *ERCIM Constraints* coordinated by F. Fages (INRIA project Contraintes, Rocquencourt) and *Programming Languages Technology* coordinated by N. Jones (Diku, Copenhague).

Olivier Bournez, Emmanuel Hainry and Paulin de Naurois are involved in the “Computability in Europe” network. This network aims at federating research efforts in Europe of teams working on computability and complexity, and in particular on new paradigms of computation such as analog computations.

We are involved in the thematic network APPSEM II on “Applied Semantics”, which is coordinated by Martin Wirsing (LMU, Muenchen). This network is funded by the “5th Framework Programme” (FP5). We contribute to the following themes of the network: proofs assistants, functional programming and dependent types; types and type inference in programming; resource models and web data (e.g. resource-bounded computation reasoning about linked data); continuous phenomena in Computer Science (e.g. computing with real numbers).

We participate to REWERSE - “Reasoning on the Web”, a Network of Excellence (NoE) within the “6th Framework Programme” (FP6), Information Society Technologies (IST). The main objective of this project is the development of a coherent and complete, yet minimal, collection of inter-operable reasoning languages for advanced Web systems and applications. These languages will be tested on context-adaptive Web systems and Web-based decision support systems selected as test-beds for proof-of-concept purposes. Finally, we aim at bringing the proposed languages to the level of open pre-standards amenable to submissions to standardization bodies such as the W3C.

8.5. International bilateral initiatives

Chili. Since 2002, we have a French-Chilean cooperation with the Federico Santa Maria Technical University of Valparaiso. This project, called COCARS and supported by CONICYT and INRIA, is about the use of rules and strategies for the design of constraint solvers.

Lisbon. Olivier Bournez and Emmanuel Hainry cooperate with Manuel Campagnolo and Daniel Graça from Lisbon, under the framework of a PAI Pessoa Grant.

London. Clara Bertolissi, Horatiu Cirstea, Germain Faure and Claude Kirchner cooperate with Maribel Fernandez and Ian Mackie from King’s College (London) and with Gilles Dowek, Jean-Pierre Jouannaud and François-Régis Sinot from LIX-École Polytechnique (Paris), under the framework of a PAI Alliance Grant.

Brazil. Project INRIA-CNPq (Brazil), DA CAPO - Automated deduction for the verification of specifications and programs. It is a project on the development of proof systems for the verification of specifications and software components. The coordinators of this project are David Déharbe (UFRN Natal, Brazil) and Christophe Ringeissen (CASSIS). On the french side, DA CAPO also involves the CASSIS project.

8.6. Invited lecturers

The program of the seminars is available at http://protheo.loria.fr/seminaires_en.html.

– Alain Frish (Projet Cristal, INRIA), "CDuce: un langage fonctionnel avec types et motifs XML".
– Alain Frish (Projet Cristal, INRIA), "Sous-typage ensembliste avec types flèche et combinaisons booléennes".
– Yves Guiraud (Équipe Logique de la Programmation, IML), "Polygraphes, réécriture et logique".
– Isabelle de Lamberterie (CNRS), "Questions juridiques liées à la pérénisation de la société de l’information".
– Christophe Alias (Équipe Compsys, LIP), "Optimisation de programmes par reconnaissance de templates".
– Pierre-Yves Strub (LIX), "Le calcul des constructions congruentes".
– Florent Kirchner (LIX), "A proof monad".
– Arnaud Spiwack (ENS Cachan), "Reducibility domain".
– Dan Dougherty (WPI), "Policy-informed program analyses".
– Dan Dougherty (WPI), "Proving strong normalization for symmetric calculi".

9. Dissemination

9.1. Leadership within scientific community

AFIT	French chapter of EATCS
ASIAN	Asian Computing Science Conference
CSL	Conference of the European Association for Computer Science Logic
DCM	Workshop on Developments in Computational Models
GTTSE	Summer School on Generative and Transformational Techniques in Software Engineering
IEHSC	International Embedded and Hybrid Systems Conference
IFIP	International Federation for Information Processing
IFIP WG	IFIP Working Group
JFLA	French-speaking workshop on Applicative Languages
LDTA	Language Descriptions, Tools and Applications
LICS	International Conference on Logics in Computer Science
LPAR	International Conference on Logic for Programming Artificial Intelligence and Reasoning
PPDP	International Conference on Principles and Practice of Declarative Programming
QPQ	Online journal for peer-reviewed source code for deductive software components
RTA	International Conference on Rewriting Techniques and Applications
RULE	International Workshop on Rule-Based Programming
SPECIF	French society of professors and researchers in computer science
STACS	Symposium on Theoretical Aspects of Computer Science

- Olivier Bournez:
  - Leader of the French node of the “Computability in Europe” network.
  - Leader of an ARASSIA on applications of game theory to security of algorithms for networks.
  - Leader of a PAI PESSOA with Manuel Campagnolo in Lisbon.
  - Member of the board of directors of AFIF (french chapter of European Association for Theoretical Computer Science).
  - Member of the program committee of CIE 2006.
  - Organization in Paris II University of a Day about “Algorithmic Game Theory. Applications to Computer and Sensor Networks” on May 16th 2006.
- Horatiu Cirstea
  - Leader of a PAI Alliance with Maribel Fernandez in London.
  - Program committees of Rule 2006, Secret 2006, DCM 2006.
- Isabelle Gnaedig:
  - Member of the QSL coordination board.
  - Coordinator of the Protheo part of the ACISI project Modulogic.
  - Manager of the PRST-IL project Toundra.
- Claude Kirchner:

- Chair of the scientific committee for the national ACISI programs
- Chair of the evaluation committee of the ANR SETIN2006 program.
- Chair of the LORIA building extension committee.
- Editorial boards of *Journal of Automated Reasoning*, *Journal of Information Science and Engineering*, *Journal of Applied Logic*.
- Invited editor of the Information Security section of Vuibert's encyclopedia of Informatics and Information Systems [20].
- Program committees of WRLA2006: 6th International Workshop on Rewriting Logic and its Applications, co-chair of SecReT 2006: 1st International Workshop on Security and Rewriting Techniques, co-chair of the third international workshop on the rewriting calculus, LSFA'06 Brazilian Workshop on Logical and Semantic Frameworks, with Applications, Chair of the scientific committee of the first international school on Rewriting (ISR'2006).
- Chair of the IFIP WG 1.6 on rewriting and applications.
- Chair of the PaRI-STIC conference (Nancy, 22-24 November).
- Member of the advisory boards of LICS and PPDP.
- Co-organizer of the "Symposium Franco-Japonais sur la sécurité informatique " (Tokyo, December 4, 5)
- Member of the working group on research and perspectives of the CISSI.

- Hélène Kirchner:
  - Director of LORIA and INRIA Lorraine.
  - Coordinator of the regional Research and Technology Project on Software Intelligence (Pôle de Recherche Scientifique et Technologique Intelligence Logicielle, PRST-IL)
  - Editorial boards of *Annals of Mathematics and Artificial Intelligence*, *Computing and Informatics* and *Logical Methods in Computer Science*.
  - Editorial board of the QPQ forum on rewriting.
  - Program committees of IJCAR'06 and of the Symposium on Artificial Intelligence and Mathematics 2006.
  - Member of Scientific directorate of the Dagstuhl international conference center.
- Pierre-Etienne Moreau:
  - LORIA committee for computer and software equipments.
  - Member of the organizing committee of ISR 2006.
  - Program committee of RULE 2006, WRLA 2006, and JFPC 2006.
  - Co-chair with Thérèse Hardin of JFLA'06.
  - Steering committee of LDTA.

9.2. Teaching

We do not mention the teaching activities of the various teaching assistants and lecturers of the project who work in various universities of the region.

- Olivier Bournez:
  - M2 course on algorithmic verification at UHP.
  - M2 course on semantic of parallel and distributed systems at UHP
  - M1 course on algorithmic and computational complexity at UHP.

- Horatiu Cirstea:
  - M1 course on rule based programming.
- Isabelle Gnaedig:
  - Coordinator of the course on program and software specification at ESIAL.
  - Courses on algebraic specifications, the LOTOS language and the semantic of concurrent processes at ESIAL.
- Claude Kirchner:
  - Master course (M2) in Nancy on programming and proving with rule based languages, with Pierre-Etienne Moreau.
  - Master course (M1) in Nancy on programming and proving with rule based languages, with Pierre-Etienne Moreau and Horatiu Cirstea.
  - Lecture at Science-Po.
- Pierre-Etienne Moreau:
  - Master course in Nancy on programming and proving with rule based languages, with Claude Kirchner.
  - Lectures at ESIAL on fundamental data-structures.
  - Java courses at IUT-Nancy2
  - Lecture at ISR (International School on Rewriting)

9.3. Invited talks

- Olivier Bournez:
  - Invited Talk “Comparing the power of several Analog models” at “Journées Arithmétiques Faibles, Complexité, Modèles Finis, Bases de Données”, Clermont-Ferrand, June 21th 2006.
  - Invited Talk “Comparing the power of several Analog models” at workshop “Continuous Dynamics and Computability”, ENS Paris. May 3rd 2006.
  - Invited Talk “Syntactic Characterizations of Some Complexity Classes in Blum/Shub/Smale Model”at IST, Lisbon, March 10th, 2006
- Claude Kirchner: Invited talks
  - Delhi and Bangalore, week 6: Indo-French Workshop in Computer Science: “Domain Specific Deduction”
  - Pekin, week 12: Université de Tsinghua: “Rewriting Techniques and Applications”, Chinese Academy of Sciences: “Rewrite based computation and deduction”, Chinese Academy of Sciences: “Domain Specific Deduction”.
  - San Diego, week 26: Festschrift in honor of Joseph Goguen: “Completion is an Instance of Abstract Canonical System Inference”.
  - Seattle: week 32: Rule workshop: “Rewriting (your) Calculus”.
  - Natal (Brasil), week 38: Brazilian Symposium on Formal Methods: “Logical aspects of the rewriting calculus”.
  - London, week 43: London Theoretical Computer Science Seminar: “Computational and Deductive Aspects of the Rewriting Calculus”.
  - Tokyo, week 49: The 2nd Franco-Japanese Computer Security Workshop: “Super Deduction Modulo”.

- Pierre-Etienne Moreau:
  - “Transformation de documents par filtrage”, IRIT, Toulouse, February 2006.

9.4. Visits

- Olivier Bournez
  - One week at the IST of Lisbon (Portugal) in March 2006.
- Claude Kirchner
  - One week in Delhi and Bangalore (India) visiting IIT Delhi, and Indian Institut of Sciences and Philips’s labs at Bangalore.
  - One week in Beijing (China) visiting LIAMA, Tsinghua university and the Academy of sciences (March).
- Hélène Kirchner
  - One week in Beijing (China) visiting LIAMA, Tsinghua university and the Academy of sciences (March).

9.5. Thesis and admission committees

- Olivier Bournez:
  - AFIT PhD thesis award committee.
  - UHP recruitment committee (section 27), substitute.
  - INPL recruitment commitee (section 27), substitute.
  - Metz University recruitment committee (section 27), substitute.
- Frédéric Blanqui:
  - Secretary of the SPECIF PhD thesis award committee.
- Claude Kirchner:
  - Luc Bouganim “Sécurisation du contrôle d’accès dans les bases de données” HDR (member),
  - Diana Toma “Vérification formelle des systèmes digitaux par démonstration de théorèmes: application aux composants crytographiques” PhD (referee),
  - Anne Canteau “Analyse et conception de chiffrements à clef secrète”, HDR (member),
  - Pascal Lafourcade “Vérification des protocoles cryptographiques en présence de théories équationnelles”, PhD, (président),
  - Richard Bonichon “Tableaux et déduction modulo” PhD (referee),
  - Ozan Kahramanoğulları“Nondeterminism and Language Design in Deep Inference” (referee),
  - Antoine Reilles “Réécriture et compilation de confiance” PhD (member).
- Hélène Kirchner:
  - Member of recruitment committees (section 27) of UHP Nancy1, Nancy2, INPL.
- Pierre-Etienne Moreau:
  - Antoine Reilles “Réécriture et compilation de confiance”, PhD
  - Member of the UHP recruitment committee (section 27).
  - Participation to an INRIA recruitment committee for research engineers.

10. Bibliography

Major publications by the team in recent years

[1] G. BARTHE, H. CIRSTEA, C. KIRCHNER, L. LIQUORI. *Pure Patterns Type Systems*, in " Principles of Programming Languages - POPL2003, New Orleans, USA", ACM, Jan 2003, p. 250–261.

[2] F. BLANQUI. *Definitions by Rewriting in the Calculus of Constructions*, in "Mathematical Structures in Computer Science", vol. 15, n^{o} 1, Feb 2005, p. 37-92.

[3] O. BOURNEZ, E. HAINRY. *An analog Characterization of Elementarily Computable Functions Over the Real Numbers*, in "31st International Colloqiium on Automata, Languages and Programming - ICALP'2004, Turku, Finland", J. DIAZ, J. KARHUMÄKI, A. LEPISTO, D. T. SANNELLA (editors). , Lecture Notes in Computer Science, vol. 3142, Springer, Jul 2004, p. 269-280.

[4] O. BOURNEZ, E. HAINRY. *Real Recursive Functions and Real Extensions of Recursive Functions*, in "Machines and Universal Computations - MCU'2004, Saint-Petersburg, Russia", Sep 2004.

[5] H. CIRSTEA, C. KIRCHNER. *The rewriting calculus - Part I and II*, in "Logic Journal of the Interest Group in Pure and Applied Logics", vol. 9, n^{o} 3, May 2001, p. 427-498.

[6] G. DOWEK, T. HARDIN, C. KIRCHNER. *Theorem Proving Modulo*, in "Journal of Automated Reasoning", vol. 31, n^{o} 1, Nov 2003, p. 33-72.

[7] O. FISSORE, I. GNAEDIG, H. KIRCHNER. *A proof of weak termination providing the right way to terminate*, in "1st International Colloquium on THEORETICAL ASPECTS OF COMPUTING, Guiyang, China", LNCS, vol. 3407, Springer Verlag, September 2004, p. 356-371.

[8] H. KIRCHNER, P.-E. MOREAU. *Promoting Rewriting to a Programming Language : A Compiler for Non-Deterministic Rewrite Programs in Associative-Commutative Theories*, in "Journal of Functional Programming", vol. 11, n^{o} 3, Mar 2001, p. 207-251.

[9] P.-E. MOREAU, C. RINGEISSEN, M. VITTEK. *A Pattern Matching Compiler for Multiple Target Languages*, in "12th Conference on Compiler Construction, Warsaw (Poland)", G. HEDIN (editor). , LNCS, vol. 2622, Springer-Verlag, May 2003, p. 61–76.

Year Publications

Doctoral dissertations and Habilitation theses

[10] O. BOURNEZ. *Modèles Continus. Calculs. Algorithmique Distribuée*, HDR, INPL, Nancy (France), 7 Décembre 2006.

[11] E. HAINRY. *Modèles de calculs sur les réels. Résultats de Comparaisons*, Ph. D. Thesis, Université Henry Poincaré, Nancy (France), 7 Décembre 2006.

[12] A. REILLES. *Réécriture et compilation de confiance*, Ph. D. Thesis, Institut National Polytechnique de Lorraine, Nancy (France), 27 Novembre 2006.

Articles in refereed journals and book chapters

[13] P. BALDAN, C. BERTOLISSI, H. CIRSTEA, C. KIRCHNER. *A rewriting calculus for cyclic higher-order term graphs*, in "Mathematical Structures in Computer Science", 2006, https://hal.inria.fr/inria-00110872/en/.

[14] O. BOURNEZ, M. L. CAMPAGNOLO. *New Computational Paradigms*, B. COOPER (editor). , chap. A Survey on Continuous Time Computation, Springer, 2006, https://hal.inria.fr/inria-00102948/en/.

[15] O. BOURNEZ, M. CAMPAGNOLO, D. GRAÇA, E. HAINRY. *Polynomial differential equation compute all real computable functions*, in "Journal of Complexity", 2006, https://hal.inria.fr/inria-00102947/en/.

[16] H. CIRSTEA, C. BERTOLISSI, C. KIRCHNER. *Expressing Combinatory Reduction Systems Derivations in the Rewriting Calculus*, in "Higher-Order and Symbolic Computation", 2006, https://hal.inria.fr/inria-00110869/en/.

[17] H. CIRSTEA, G. FAURE, C. KIRCHNER. *A Rho-Calculus of explicit constraint application*, in "Higher-Order and Symbolic Computation", vol. 20, 2006, https://hal.inria.fr/inria-00000628/en/.

[18] N. DERSHOWITZ, C. KIRCHNER. *Abstract canonical presentations, Theoretical Computer Science*, in "Theoretical Computer Science", vol. 357, n^{o} 1-3, 2006, p. 53–69, http://hal.inria.fr/inria-00121760/en/.

[19] I. GNAEDIG, H. KIRCHNER. *Termination of rewriting under strategies: a generic approach*, in "ACM Transactions on Computational Logic", 2006, https://hal.inria.fr/inria-00113156/en/.

[20] C. KIRCHNER. *Encyclopédie de l'informatique et des systèmes d'information*, J. AKOKA, I. COMYN-WATTIAU (editors). , chap. Sécurité Informatique: Introduction, Vuibert, 2006, http://hal.inria.fr/inria-00121763/en/.

Publications in Conferences and Workshops

[21] O. ANDREI, L. IBANESCU, H. KIRCHNER. *Non-intrusive formal methods and strategic rewriting for a chemical application*, in "Algebra, Meaning, and Computation: A Festschrift Symposium in Honor of Joseph Goguen, 27/06/2006, San Diego, USA", K. FUTATSUGI, J.-P. JOUANNAUD, J. MESEGUER (editors). , in: Lecture Notes in Computer Science, Algebra, Meaning, and Computation, Essays Dedicated to Joseph A. Goguen on the Occasion of His 65th Birthday, vol. 4060, Springer Verlag, 2006, p. 194-215, https://hal.inria.fr/inria-00115521/en/.

[22] E. BALLAND, C. KIRCHNER, P.-E. MOREAU. *Formal Islands*, in "11th International Conference on Algebraic Methodology and Software Technology - AMAST '06, 05/07/2006, Kuressaare/Estonia", 2006, https://hal.inria.fr/inria-00001146/en/.

[23] E. BALLAND, C. KIRCHNER, P.-E. MOREAU, A. SANTANA DE OLIVEIRA. *Modular Formal Islands: Embed theory in your practice*, in "Third Taiwanese-French Conference on Information Technology - TFIT 2006, 28/03/2006, Nancy/France", INRIA/LORIA, 2006, https://hal.inria.fr/inria-00001186/en/.

[24] E. BALLAND, P.-E. MOREAU. *Optimizing pattern matching compilation by program transformation*, in "3rd Workshop on Software Evolution through Transformations SeTra 2006, 17/09/2006, Natal, Rio Grande do Norte, Brazil", 2006, https://hal.inria.fr/inria-00000763/en/.

[25] C. BERTOLISSI, C. KIRCHNER. *The Rewriting Calculus as a Combinatory Reduction System*, in "Prooceedings the FoSSaCS, 26/03/2007, Braga, Portugal", H. SEIDL (editor). , lncs, Springer-Verlag, 2006, http://hal.inria.fr/inria-00121792/en/.

[26] F. BLANQUI. *Higher-order dependency pairs*, in "Eighth International Workshop on Termination - WST 2006, 15/08/2006, Seattle, Washington/USA", 2006, https://hal.inria.fr/inria-00084821/en/.

[27] F. BLANQUI, S. COUPET-GRIMAL, W. DELOBEL, S. HINDERER, A. KOPROWSKI. *CoLoR: a Coq library on rewriting and termination*, in "Eighth International Workshop on Termination - WST 2006, 15/08/2006, Seattle, Washington/USA", 2006, https://hal.inria.fr/inria-00084835/en/.

[28] F. BLANQUI, T. HARDIN, P. WEIS. *On the implementation of construction functions for non-free concrete data types*, in "16th European Symposium on Programming, 24/03/2007, Braga (Portugal)", 2006, http://hal.inria.fr/inria-00095110/en/.

[29] F. BLANQUI, J.-P. JOUANNAUD, A. RUBIO. *Higher-Order Termination: from Kruskal to Computability*, in "13th International Conference on Logic for Programming, Artificial Intelligence and Reasoning - LPAR 2006, 13/11/2006, Phnom Penh/Cambodge", LNCS, vol. 4246, 2006, https://hal.inria.fr/inria-00091308/en/.

[30] F. BLANQUI, C. KIRCHNER, C. RIBA. *On the confluence of lambda-calculus with conditional rewriting*, in "Foundations of Software Science and Computation Structures, 31/03/2006, Vienne/Autriche", Lecture Notes in Computer Science, n^{o} 3921, 2006, https://hal.inria.fr/inria-00000729/en/.

[31] F. BLANQUI, C. RIBA. *Combining typing and size constraints for checking the termination of higher-order conditional rewrite systems*, in "13th International Conference on Logic for Programming, Artificial Intelligence and Reasoning - LPAR 2006, 13/11/2006, Phnom Penh/Cambodge", LNCS, vol. 4246, 2006, https://hal.inria.fr/inria-00084837/en/.

[32] O. BOURNEZ, M. LAMEIRAS CAMPAGNOLO, D. GRAÇA, E. HAINRY. *The General Purpose Analog Computer and Computable Analysis are Two Equivalent Paradigms of Analog Computation*, in "3rd International Conference on Theory and Applications of Models of Computation - TAMC'2006, 2006, Pekin/Chine", 2006, https://hal.inria.fr/inria-00102946/en/.

[33] G. BUREL, C. KIRCHNER. *Completion is an Instance of Abstract Canonical System Inference*, in "Algebra, Meaning, and Computation: A Festschrift Symposium in Honor of Joseph Goguen, 06/2006, San Diego/USA", K. FUTATSUGI, J.-P. JOUANNAUD, J. MESEGUER (editors). , Lecture Notes in Computer Science, vol. 4060, Springer Verlag, 2006, p. 497-520, https://hal.inria.fr/inria-00000775/en/.

[34] G. BUREL, C. KIRCHNER. *Cut Elimination in Deduction Modulo by Abstract Completion*, in "FoSSaCS 2007, 24/03/2007, Braga/Portugal", 2006, https://hal.inria.fr/inria-00115556/en/.

[35] H. CIRSTEA, G. FAURE, M. FERNÁNDEZ, I. MACKIE, F.-R. SINOT. *From functional programs to interaction nets via the Rewriting Calculus*, in "Sixth International Workshop on Reduction Strategies in Rewriting and Programming, 11/08/2006, Seattle, Washington", 2006, https://hal.inria.fr/inria-00000821/en/.

[36] H. CIRSTEA, C. HOUTMANN, B. WACK. *Distributive rewriting calculus*, in "Sixth International Workshop on Reduction Strategies in Rewriting and Programming, 01/04/2006, Vienna/Austria", 2006, https://hal.inria.fr/inria-00110867/en/.

[37] G. FAURE. *Matching modulo superdeveloppements. Application to second-order matching.*, in "13th International Conference on Logic for Programming Artificial Intelligence and Reasoning - LPAR 2006, 13/11/2006, Phnom Penh/Cambodge", 2006, https://hal.inria.fr/inria-00095608/en/.

[38] G. FAURE. *Term collection in lambda/rho-calculi*, in "2nd International Workshop on Developments in Computational Models - DCM 2006, 16/07/2006, Venise/Italie", 2006, https://hal.inria.fr/inria-00102993/en/.

[39] G. FAURE, A. REILLES. *Tom llustrated on an implementation of the explicit rewriting calculus*, in "Workshop on Rewriting Techniques and Applications, 01/04/2006, Vienna /Austria", 2006, https://hal.inria.fr/inria-00096026/en/.

[40] I. GNAEDIG, H. KIRCHNER. *Computing Constructor Forms with Non Terminating Rewrite Programs*, in "Symposium on Principles and Practice of Declarative Programming - PPDP'06, 10/07/2006, Venise/Italie", vol. Proceedings of the Eighth ACM-SIGPLAN International Symposium on Principles and Practice of Declarative Programming, ACM, 2006, https://hal.inria.fr/inria-00112083/en/.

[41] B. OLIVIER, G. FLORENT. *Proving Positive Almost Sure Termination Under Strategies*, in "17th International Conference on Rewriting Techniques and Applications - RTA'2006, 12/08/2006, Seattle, WA/USA", F. PFENNING (editor). , Lecture Notes in Computer Science, vol. 4098, Springer, 2006, p. 357–371, https://hal.inria.fr/inria-00102945/en/.

[42] A. REILLES. *Canonical Abstract Syntax Trees*, in "Workshop on Rewriting Techniques and Applications, 0000, Vienna /Austria", Carolyn Talcott and Grit Denker, 2006, https://hal.inria.fr/inria-00000967/en/.

[43] A. SANTANA DE OLIVEIRA. *Rewriting-Based Access Control Policies*, in "1st International Workshop on Security and Rewriting Techniques - SecReT 2006, 16/09/2006, Venice/Italy", M. FERNÁNDEZ, C. KIRCHNER (editors). , Maribel Fernández, 2006, https://hal.inria.fr/inria-00112340/en/.

Internal Reports

[44] F. BLANQUI. *(HO)RPO Revisited*, Rapport de recherche INRIA, n^{o} RR-5972, 2006, https://hal.inria.fr/inria-00090488/en/.

Miscellaneous

[45] É. BALLAND, P. BRAUNER, R. KOPETZ, P.-É. MOREAU, A. REILLES. *The Tom Manual*, 2006, http://tom.loria.fr/soft/release-2.4/manual-2.4/index.html.

[46] P. BRAUNER. *Un Calcul des Séquents Extensible*, Master Thesis, Université Henri Poincaré – Nancy 1, jun 2006.

[47] C. HOUTMANN. *Cohérence de la Déduction Surnaturelle*, Master Thesis, École Normale Supérieure de Cachan, sep 2006.

References in notes

[48] T. ARTS, J. GIESL. *Termination of Term Rewriting Using Dependency Pairs*, in "Theoretical Computer Science", vol. 236, 2000, p. 133-178.

[49] C. BERTOLISSI, P. BALDAN, H. CIRSTEA, C. KIRCHNER. *A rewriting calculus for cyclic higher-order term graphs*, in "2nd International Workshop on Term Graph Rewriting - TERMGRAPH'2004, Rome, Italy", M. FERNÁNDEZ (editor). , Electronic Notes in Theoretical Computer Science, Oct 2004.

[50] C. BERTOLISSI. *The graph rewriting calculus : confluence and expressiveness*, in "9th Italian conference on Italian Conference on Theoretical Computer Science - ICTCS 2005, Siena, Italy", G. M. P. MARIO COPPO (editor). , Lecture Notes in Computer Science, vol. 3701, Springer Verlag, Oct 2005, p. 113–127.

[51] C. BERTOLISSI, H. CIRSTEA, C. KIRCHNER. *Translating Combinatory Reduction Systems into the Rewriting Calculus*, in "4th International Workshop on Rule-Based Programming (RULE 2003), Valencia, Spain", Long version, Jun 2003, http://www.loria.fr/publications/2003/A03-R-057/A03-R-057.ps.

[52] F. BLANQUI. *Definitions by rewriting in the Calculus of Constructions*, in "Mathematical Structures in Computer Science", Journal version of LICS'01, vol. 15, n^{o} 1, 2005, p. 37-92, http://hal.inria.fr/inria-00105648/en/.

[53] G. BOUDOL. *Lambda-Calculi for (Strict) Parallel Functions*, in "Inf. Comput", vol. 108, n^{o} 1, January 1994.

[54] A. BURRONI. *Higher-dimensional word problems with applications to equational logic*, in "Theoretical Computer Science", vol. 115, n^{o} 1, July 1993, p. 43-62.

[55] H. CIRSTEA, G. FAURE, C. KIRCHNER. *A rho-calculus of explicit constraint application*, in "Workshop on Rewriting Logic and Applications, Barcelona (Spain)", Electronic Notes in Theoretical Computer Science, Mar 2004.

[56] H. CIRSTEA, L. LIQUORI, B. WACK. *Rewriting Calculus with Fixpoints: Untyped and First-order Systems*, in "Types for Proofs and Programs (TYPES), Torino (Italy)", S. BERARDI, M. COPPO, F. DAMIAN (editors). , Lecture Notes in Computer Science, vol. 3085, May 2003, p. 147–171.

[57] H. CIRSTEA, P.-E. MOREAU, A. REILLES. *Rule based programming in Java for protocol verification*, in "Proceedings of the 5th International Workshop on Rewriting Logic and its Applications, WRLA'2004 (Barcelona, Spain), Barcelona (Spain)", N. MARTI-OLIET (editor). , vol. 117, Electronic Notes in Theoretical Computer Science, April 2004, p. 209–227, http://www.loria.fr/~moreau/Papers/CirsteaMR-WRLA2004.pdf.

[58] G. COLATA. *With Major Math Proof, Brute Computers Show Flash of Reasoning Power*, in "The New York Times", Tuesday December 10, 1996.

[59] COQ-DEVELOPMENT-TEAM. *The Coq Proof Assistant Reference Manual - Version 8.0*, 2004, http://coq.inria.fr/, INRIA Rocquencourt, France.

[60] S. COUPET-GRIMAL, W. DELOBEL. *An Effective Proof of the Well-Foundedness of the Multiset Path Ordering*, in "Applicable Algebra in Engineering Communication and Computing", To appear, 2006.

[61] E. DEPLAGNE, C. KIRCHNER, H. KIRCHNER, Q.-H. NGUYEN. *Proof Search and Proof Check for Equational and Inductive Theorems*, in "Proceedings of CADE-19, Miami, Florida", F. BAADER (editor). , Lecture Notes in Computer Science, Springer-Verlag, July 2003.

[62] N. DERSHOWITZ, C. KIRCHNER. *Abstract Saturation-based Inference*, in "Eighteenth Annual IEEE Symposium on Logic in Computer Science - LICS'2003), Ottawa, Canada", Jun 2003, http://www.loria.fr/publications/2003/A03-R-422/A03-R-422.ps.

[63] D. DOUGHERTY. *Adding Algebraic Rewriting to the Untyped Lambda Calculus*, in "Information and Computation", vol. 101, n^{o} 2, 1992, p. 251-267.

[64] G. DOWEK, T. HARDIN, C. KIRCHNER. *Theorem Proving Modulo*, Rapport de Recherche, n^{o} 3400, Institut National de Recherche en Informatique et en Automatique, Apr 1998, http://hal.inria.fr/inria-00077199.

[65] M. FERNÁNDEZ, I. MACKIE, 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.

[66] O. FISSORE, I. GNAEDIG, H. KIRCHNER. *CARIBOO : An induction based proof tool for termination with strategies*, in "Proceedings of the Fourth International Conference on Principles and Practice of Declarative Programming, Pittsburgh (USA)", ACM Press, Oct 2002, p. 62–73.

[67] Y. GUIRAUD. *Termination orders for three-dimensional rewriting*, in "Journal of Pure and Applied Algebra", vol. 207, n^{o} 2, October 2006, p. 341-371.

[68] Y. GUIRAUD. *The three dimensions of proofs*, in "Annals of Pure and Applied Logic", vol. 141, n^{o} 1-2, August 2006, p. 266-295.

[69] O. HERMANT. *Semantic Cut Elimination in the Intuitionistic Sequent Calculus.*, in "TLCA", P. URZYCZYN (editor). , LNCS, vol. 3461, Springer-Verlag, 2005, p. 221-233.

[70] L. IBANESCU. *Programmation par règles et stratégies pour la génération automatique de mécanismes de combustion d'hydrocarbures polycycliques*, Thèse d'université, Institut National Polytechnique de Lorraine - INPL, France, 2004.

[71] J.-P. JOUANNAUD, C. KIRCHNER. *Solving equations in abstract algebras: a rule-based survey of unification*, in "Computational Logic. Essays in honor of Alan Robinson, Cambridge (MA, USA)", J.-L. LASSEZ, G. PLOTKIN (editors). , chap. 8, The MIT press, 1991, p. 257-321.

[72] J.-P. JOUANNAUD, A. RUBIO. *The Higher-Order Recursive Path Ordering*, in "Proceedings of LICS", 1999.

[73] O. KAHRAMANOĞULLARI, P.-E. MOREAU, A. REILLES. *Implementing Deep Inference in TOM*, in "Structures and Deduction, Lisbon, Portugal", P. BRUSCOLI, FRANÇOIS. LAMARCHE, C. STEWART (editors). , ISSN 1430-211X, Technische Universität Dresden, July 2005, p. 158–172, http://www.loria.fr/~moreau/Papers/KahramanogullariMR-SD2005.pdf.

[74] C. KIRCHNER, H. K. KIRCHNER, F. NAHON. *Narrowing Based Inductive Proof Search*, June 2005, Presented at the Workshop on Programming Logics in memory of Harald Ganzinger.

[75] C. KIRCHNER, H. KIRCHNER, A. SANTANA DE OLIVEIRA. *Anchoring modularity in HTML.*, in "1st International Workshop on Automated Specification and Verification of Web Sites - WWV'05, 14/03/2005, Valencia/Spain", 2005, p. 139-151, http://hal.inria.fr/inria-00000750/en/.

[76] C. KIRCHNER, R. KOPETZ, P. MOREAU. *Anti-Pattern Matching*, in "Proceedings of ESOP", 2007.

[77] D. E. KNUTH, P. B. BENDIX. *Simple word problems in universal algebras*, in "Computational Problems in Abstract Algebra, Oxford", J. LEECH (editor). , Pergamon Press, 1970, p. 263-297.

[78] A. KOPROWSKI. *Certified Higher-Order Recursive Path Ordering*, in "Proceedings of RTA'06".

[79] Y. LAFONT. *Interaction Nets*, in "Proceedings of the 17th ACM Symposium on Principles of Programming Languages (POPL'90)", ACM Press, January 1990, p. 95–108.

[80] C. MARCHÉ. *Normalized Rewriting: An Alternative to Rewriting Modulo a Set of Equations*, in "Journal Symb. Comput", vol. 21, n^o 3, 1996.

[81] W. MCCUNE. *Solution of the Robbins Problem*, in "JAR", vol. 19, n^o 3, 1997, p. 263-276.

[82] J. MESEGUER. *Conditional rewriting logic as a unified model of concurrency*, in "TCS", vol. 96, n^o 1, 1992, p. 73-155.

[83] C. MOORE. *Recursion Theory on the Reals and Continuous-time Computation*, in "Theoretical Computer Science", vol. 162, 1996, p. 23-44.

[84] P.-E. MOREAU, C. RINGEISSEN, M. VITTEK. *A Pattern Matching Compiler for Multiple Target Languages*, in "12th Conference on Compiler Construction, Warsaw (Poland)", G. HEDIN (editor). , LNCS, vol. 2622, Springer-Verlag, May 2003, p. 61–76.

[85] F. MÜLLER. *Confluence of the lambda calculus with left-linear algebraic rewriting*, in "Information Processing Letters", vol. 41, n^o 6, 1992, p. 293-299.

[86] TERESE. *Term Rewriting Systems*, M. Bezem, J. W. Klop and R. de Vrijer, eds., Cambridge University Press, 2002.

[87] B. WACK. *Typage et déduction dans le calcul de réécriture*, Thèse de Doctorat, Université Henri Poincaré - Nancy I, Oct 2005.

[88] O. DE MOOR, G. SITTAMPALAM. *Higher-order Matching for Program Transformation*, in "Theoretical Computer Science", vol. 269, 2001.

[89] F. VAN RAAMSDONK. *Confluence and superdevelopments*, in "Rewriting Techniques and Applications", 1993.