INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Project-Team PROTHEO

Constraints, Mechanized Deduction and Proofs of Software Properties

Nancy - Grand Est

THEME SYM

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).*

Head of project-team
Claude Kirchner [DR INRIA, HdR]

Administrative assistant
Chantal Llorens [CNRS]

Research scientist
Frédéric Blanqui [CR INRIA]
Horatiu Cirstea [MC Nancy2]
Yves Guiraud [CR INRIA since 01/09/07]
Hélène Kirchner [DR CNRS, seconded to INRIA from February 1st, part time, HdR]
Pierre-Etienne Moreau [CR INRIA]

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

Postdoctoral fellow
Yohan Boichut [INRIA since 21/09/07]
Richard Bonichon [INRIA since 01/09/07]
Florent Garnier [Nancy 2 half-time ATER since 17/09/07]
Yves Guiraud [INRIA until 31/08/07]

Ph.D. students
Oana Andrei [CORDI]
Émilie Balland [MESR]
Paul Brauner [MESR]
Guillaume Burel [MESR]
Germain Faure [MESR until 31/08/2007]
Florent Garnier [Nancy 2 half-time ATER until 17/09/07]
Clément Houtmann [ENS Cachan followed by MESR]
Radu Kopetz [CORDI]
Fabrice Nahon [High school teacher, Ph.D defense on 26/10/07]
Antoine Reilles [INPL ATER until 31/08/07]
Colin Riba [MESR until 31/10/07]
Cody Roux [CORDI since 01/10/07]
Anderson Santana [Brazil]
Claudia Tavares [Brazil since 01/11/07]

Technical staff
Éric Deplagne [INRIA until 31/01/07]

Internships
Jonathan Demange [UHP 09/05/07–10/06/07]
Léo Ducas [ENS Ulm 01/07/07–31/08/07]
Martin Grandcolas [UHP 15/06/07–15/08/07]
Yassine Guebbas [INPL 24/09/07–27/02/08]
Pauline Kouzmenko [INPL 15/10/06–28/02/07]
Laura Lowenthal [CORDI 15/10/07–15/02/08]
Aurélien Monot [INPL 24/09/07–27/02/08]
Baptiste Payan [UHP 15/06/07–15/08/07]

Eric Ke Wang [China 01/03/07–30/06/07]

2. Overall Objectives

2.1. Introduction

The PROTHEO project aims at designing and implementing 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.

2.2. Highlights of the year

- Dissemination of *Tom* is successful not only among researchers and academics, but also among industrial actors involved in the development of complex software. Two testimonies: *Tom* adopted by Business Object. Among the INRIA software developped on Gforge, *Tom* had the greatest numbers of downloads during several months (see Section 6.2).
- TPA+CoLoR+Rainbow won the first international competition on certified automated termination provers (see Section 5.2).
- We defined a linear translation of second-order arithmetic into first-order arithmetic *modulo*, i.e. the length of proofs remains the same, bringing a completely new point of view on the role of computation in Parikh's result, conjectured by Gödel: proofs in second-order arithmetic can be unboundedly shorter than in first-order (see Section 6.3.6).

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 [82].

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) [3]. 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 [72]. 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 to 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 efficient 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 [61], [79] 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: *XML transformation*, *access control policies*, *modeling*, *program transformation*, *proof of properties*, *protocol verification*, *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.

The matching capabilities of such languages offer ease of expressivity for program transformation and optimisation, or for the (safe) transformation of XML entities.

The combination of rewrite based transformations, strategies and typing provides an expressive framework and background for several application domains: we apply these techniques to the specification and verification of protocols and access control policies, and to the study biochemical applications.

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.

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.

5.2. CoLoR and Rainbow

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

Participants: Frédéric Blanqui, Léo Ducas.

CoLoR and *Rainbow* are distributed under CeCILL license on http://color.loria.fr/.

CoLoR is a *Coq* [62] library on rewriting and termination. It is intended to serve as a basis for certifying the output of automated termination provers like TPA, AProVE, Torpa. It contains libraries on:

- Mathematical structures: relations, semi-rings.
- Data structures: lists, vectors, integer polynomials with multiple variables, finite multisets, matrices.
- Term structures: strings, algebraic terms with symbols of fixed arity, algebraic terms with varyadic symbols, simply typed lambda-terms.
- Transformation techniques: conversion from strings to algebraic terms, conversion from algebraic to varyadic terms, arguments filtering, rule elimination, dependency pairs.
- Termination criteria: polynomial interpretations, multiset ordering, lexicographic ordering, first and higher order recursive path ordering, matrix interpretations, dependency graph decomposition.

This year, the *CoLoR* library was extended with semi-rings and matrix interpretations [74] and dependency graph decomposition [65].

Rainbow is a tool for automatically certifying termination proofs expressed in some termination proof grammar (TPG). 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[1]. On May 2006, Frédéric Blanqui organized the first workshop on the certification of termination proofs. It gathered eight participants from Austria, Germany, the Netherlands and France for two days.

In June, by using Adam Koprowski's automated termination prover TPA[2], which generates proofs in the *Rainbow* format, we could certify the termination of 463/1977 = 23.4% of the TRS termination problem data base (TPDB version 4.0) of the international competition on termination. In 2006, the score was 167/864 = 19.3% (TPDB version 3.2). With Leo Ducas' last development on the dependency graph decomposition, we expect to reach at least 30%.

For the first time in 2007, the annual international competition on termination organized a competition for certified provers. There were three competitors: TPA+Rainbow+CoLoR, TTT2+Rainbow+CoLoR and CiME2.99+Coccinelle, and TPA+Rainbow+CoLoR won the competition. The results can be consulted on the web site of the competition.

CoLoR was also presented by Frédéric Blanqui in an invited talk at TYPES'07 (the slides of the talk are available on the CoLoR web site).

5.3. Elan

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

Participants: Éric 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. Recently, support for integer overflow has been added.

5.4. Moca

Keywords: *completion*, *functional programming*, *non-free data types*, *rewriting*.

Participants: Frédéric Blanqui, Richard Bonichon, Laura Lowenthal.

Moca is distributed under QPL on http://moca.inria.fr/. The first release has been done on 27 April 2007.

Moca is a general construction functions generator for OCaml [64] data types with invariants.

Moca allows the high-level definition and automatic management of complex invariants for data types. In addition, *Moca* provides the automatic generation of maximally shared values, independantly or in conjunction with the declared invariants.

A relational data type is a concrete data type that declares invariants or relations that are verified by its constructors. For each relational data type definition, *Moca* compiles a set of construction functions that implements the declared relations.

[1] http://www.lri.fr/~marche/termination-competition/
[2] http://www.win.tue.nl/tpa/

Moca supports two kinds of relations:

- algebraic relations (such as associativity or commutativity of a binary constructor),
- general rewrite rules that map some pattern of constructors and variables to some arbitrary user's define expression.

Algebraic relations are primitive, so that *Moca* ensures the correctness of their treatment. By contrast, the general rewrite rules are under the programmer's responsability, so that the desired properties must be verified by a programmer's proof before compilation (including for completeness, termination, and confluence of the resulting term rewriting system).

Algebraic invariants are specified by using keywords denoting equational theories like commutativity and associativity. *Moca* generates construction functions that allow each equivalence class to be uniquely represented by their canonical value.

See Section 6.2.7 for some theoretical explanations.

5.5. Tom

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

Participants: Émilie Balland, Paul Brauner, Radu Kopetz, Pierre-Etienne Moreau, Antoine Reilles.

Since 2002, we have developed a new system called *Tom* [80], presented in [48], [20]. 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 [7]. 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 with 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.

6. New Results

6.1. Rewriting Calculus

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

The rewriting calculus, studied in our team since 1996, is a foundational framework unifying rewriting and lambda-calculus. We have now a deep understanding of its agility and properties, culminating this year with the PhD thesis of Germain Faure and refinements on the graph versions of the calculus, the relationship between the calculus and higher-order rewritings, as well as the in depth study of its typed versions.

6.1.1. Confluence of Pattern-Based Calculi

Participants: Horatiu Cirstea, Germain Faure.

Different pattern calculi integrate the functional mechanisms from the λ-calculus and the matching capabilities from rewriting. Several approaches are used to obtain the confluence but in practice the proof methods share the same structure and each variation on the way pattern-abstractions are applied needs another proof of confluence.

We have proposed in [32], [9] a generic confluence proof where the way pattern-abstractions are applied is axiomatized. Intuitively, the conditions guarantee that the matching is stable by substitution and by reduction.

Our approach directly applies to different pattern calculi, namely the lambda calculus with patterns, the pure pattern calculus and the rewriting calculus. We also characterized a class of matching algorithms and consequently of pattern-calculi that are not confluent.

6.1.2. 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 rewriting 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 [36].

In the spirit of normalized rewriting [78], the proposed approach manages term collections at the meta-level by considering only *canonical sets*, i.e. sets that are normalized for some rules (sometimes 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 Boudol [57] mainly insisted on models, we provide an operational point of view on the parallel lambda-calculus.

6.1.3. Explicit ρ-calculus

Participants: Horatiu Cirstea, Germain Faure, Claude Kirchner.

Following the works on explicit substitutions for λ-calculus, we proposed, studied and exemplified [59] 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 [13], [9]. All these calculi can be extended to arbitrary matching theories.

The explicit substitution application initially studied [59] 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 have introduced a calculus that offers support for the composition of substitutions [13]. 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 [60].

6.1.4. ρ-Calculus and Combinatory Reduction Systems

Participants: 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 with Clara Bertolissi from the University Aix-Marseille 1, the relation between ρ-calculus and higher order rewriting. We had showed how the semantics of Combinatory Reduction Systems can be expressed in terms of the rewriting calculus. The converse issue has been addressed lately: rewriting calculus derivations are simulated by Combinatory Reduction Systems derivations. As a consequence of this result, important properties, like standardisation, are deduced for the rewriting calculus [22].

6.1.5. Sharing strategy for the graph rewriting calculus

Participants: Horatiu Cirstea, Claude Kirchner.

Starting from the classical untyped ρ-calculus and in collaboration with Paolo Baldan from the University of Venice and Clara Bertolissi from the University Aix-Marseille 1, we have proposed an extension of the calculus, called *graph rewriting calculus*, handling structures containing sharing and cycles, rather than simple terms [54].

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.

We have proposed lately [18] a reduction strategy for the graph rewriting calculus which aims at maintaining the sharing information as long as possible in the terms. The corresponding reduction relation is shown to be confluent and complete with respect to the small-step semantics of the graph rewriting calculus.

6.1.6. A Rewriting Calculus for Multigraphs with Ports

Participants: Oana Andrei, Hélène Kirchner.

In [16], we defined labeled multigraphs with ports, a graph model which specifies connection points for nodes and allows multiple edges and loops. The dynamic evolution of these structures is expressed with multigraph rewrite rules and a multigraph rewriting relation. The multigraphs can be encoded using algebraic terms and multigraph rewriting is translated into term rewriting, which provides an operational semantics for the multigraph rewriting relation. This term version can be embedded in the rewriting calculus, thus defining for labeled multigraph transformations a high-level pattern calculus, called ρ_{mg}-calculus.

6.1.7. Distributive rewriting calculus

Participants: Horatiu Cirstea, Clément Houtmann.

General term rewriting systems and classical guiding strategies have been encoded in the original rewriting calculus [3] 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 [60].

In collaboration with Benjamin Wack, we have shown that the previously proposed encoding can be extended to the general case, i.e. to arbitrary term rewrite systems [34]. 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.

6.1.8. 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 [66] two alternative encodings of the ρ-calculus in interaction nets [76] are proposed (graph rewrite systems which have been used for the implementation of efficient reduction strategies for the λ-calculus). In collaboration with Maribel Fernandez (King's College), Ian Mackie (CNRS and King's college) and François-Régis Sinot (University of Porto), we showed in [33] 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.

6.1.9. Strong Normalization of Pure Pattern Type Systems

Participant: Clément Houtmann.

Pure Pattern Type Systems [53] (P^2TS) combine in a unified setting the frameworks and capabilities of rewriting and λ-calculus. Their type systems, adapted from Barendregt's λ-cube, are especially interesting from a logical point of view. Strong normalization, an essential property for logical soundness, had only been conjectured so far.

In strong collaboration with Benjamin Wack (now professeur agrégé de mathématiques), we have proved in [15] strong normalization of the simply-typed and dependently-typed P^2TS. The proof relies on a faithful translation from simply-typed P^2TS into System Fω, and then from dependently-typed P^2TS into simply-typed P^2TS.

In the untyped framework, we encoded pattern matching in the λ-calculus in a quite efficient way, ensuring that every $\rho\sigma\delta$-reduction is translated into (at least) one β-reduction. Introducing types in the translation proved an interesting challenge. One difficulty comes from the pattern matching occurring in the P^2TS types, which calls for accurate adjustments in the translation. Another remarkable point is that the typing mechanisms of even the simply-typed P^2TS can be expressed only with the expressive power of System Fω, which is rather surprising since Fω is a higher-order system featuring types depending on types.

6.2. Rule-based programming

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

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

Several improvements have been designed in pattern matching with anti-patterns, structure sharing via graph rewriting, and through powerful strategies. Also, we continue our study on applications of rule-based languages. In particular, we study their applications to XML transformations, biochemichal systems simulation, as well as the analysis and the certification of program transformations and of access control policies.

Finally, the algebraic structure of polygraph has been used as a description of graphical rule-based computations, equipped with complexity analysis tools.

6.2.1. Strategic Programming in Java

Participants: Émilie Balland, Paul Brauner, Radu Kopetz, Pierre-Etienne Moreau, Antoine Reilles.

One interest of term rewriting is its ability to describe elementary transformations. When combined with a strategy language it becomes very expressive to describe complex transformations. Starting from the *ELAN* experience, we have provided a powerful strategy language, that can be easily integrated in a Java environment. This language, which is part of the *Tom* system is presented in [20]. In [40] we present the essential feature we have considered when designing this new language based on rules and strategies. Relying on the implementation of *Tom*, we explain how these ingredients can be implemented and integrated in a Java environment.

6.2.2. 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, or to words that do not contain the letter "a".

To this end, we have defined in [38] the notion of anti-patterns and their semantics along with some of their properties. We then extended the classical notion of matching between patterns and ground terms to matching between anti-patterns and ground terms. We provided a rule-based algorithm that finds the solutions to such problems and proved 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 this is used to extend the expressiveness and usability of the *Tom* language.

A natural question that then raises is to deal with anti-pattern when some of the symbols have some equational properties. To answer this question, we generalized in [39] the syntactic anti-pattern matching to anti-pattern matching *modulo* an arbitrary equational theory E, and we study the specific and practically very useful case of associativity, possibly with a unity ($\mathcal{AU}$). To this end, based on the *syntacticness* of associativity, we present a rule-based associative matching algorithm, and we extend it to $\mathcal{AU}$. This algorithm is then used to solve $\mathcal{AU}$ anti-pattern matching problems. This allows us to be generic enough so that for instance, the *AllDiff* standard predicate of constraint programming becomes simply expressible in this framework. $\mathcal{AU}$ anti-patterns are implemented in the *Tom* language and we show some examples of their usage. An extended version of this work is available as a research report [47].

6.2.3. Bytecode rewriting

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

There exist several libraries for manipulating Java bytecode, among them BCEL and ASM are the most well-known. Although they are powerful, a deep knowledge of the API may be needed to use them effectively.

We introduce an abstraction level, based on term-rewriting, to make the definition of high-level transformations and analysis easier. Using the notion of algebraic view, we have extended the ASM library in such a way that a bytecode program can be seen as a term [21]. This gives us the possibility to directly express transformation rules without knowing the API, and thus to reduce the gap between the user's wishes and the language expressiveness. This approach can be considered similar to a domain specific language (DSL) for bytecode transformations.

6.2.4. Term graph rewriting

Participants: Émilie Balland, Paul Brauner, Claude Kirchner, Pierre-Etienne Moreau.

Program transformation and graph rewriting are strongly related. Indeed, although the structure of a program may be represented by a tree, informations about its execution like data dependencies or control flow are naturally expressed by data-structures inherently using graphs. To manage such structures, we have generalized the notion of term positions with *term paths* [19], [49]. By extending a signature with paths we obtained a new kind of rewriting called *addressed term rewriting* where terms can contain pointers.

Based on the formalization of paths and a notion of rewriting for addressed terms, we establish a simulation of term-graph rewriting by addressed terms. The main advantage of such an approach is to offer an efficient way to implement in any rule-based language term-graph rewriting features. In fact, since the simulation is completely based on standard first-order terms, this extension is non-intrusive. The integration in the *Tom* language provides a solid platform to experiment graph transformations in a concise and expressive way.

6.2.5. Rule-Based Modeling for Biochemical Applications

Participants: Oana Andrei, Hélène Kirchner.

Providing a formal description for the structure and functioning of biochemical systems, as well as formal tools for reasoning about their behavior is yet a scientific challenge. Frequently, the description of molecular complexes or chemical reactants relies on specific classes of graphs, and the interactions between the reactants involve rules applied on these classes of graphs and controlled by numerical data or specific filters. In this context of biochemical systems, typical considered problems are the exhaustive generation of all possible states of the system, the detection of specific states, or the prediction of producing specific states. In [17], we have proposed a rewriting framework for modeling molecular complexes, biochemical reaction rules, and generation of biochemical networks based on the representation of molecular complexes as a particular type of multigraphs with ports called molecular graphs. The advantage of this approach is to obtain a rewriting calculus which allows defining at the same level transformation rules and strategies for modeling rule selection and application, in order to prototype network generation.

In the biochemical model we consider, the behaviour of a protein is given by its functional domains that determine which other protein it can bind to or interact with. These domains are usually abstracted as sites that can be bound or free, visible or hidden. A protein is characterized by the collection of interaction sites on its surface. Proteins can bind to each other forming molecular complexes. Membranes can also form complexes, called tissues, due to the binding proteins on their surfaces. The structure of a complex is naturally described as an extended version of multigraphs with ports [16], and multigraph rewriting models the interactions between them. We encoded the molecular graphs as terms, the reaction patterns as rewrite rules, and the transformation on molecular graphs as a rewriting relation. We defined a rewriting calculus for molecular graphs, the ρ_{bio}-calculus, obtained from the rewrite calculus for labeled multigraphs with ports, the ρ_{mg}-calculus introduced in [16], by adding state information on ports and imposing some conditions on edges. Strategic rewriting allows modeling the control mechanism in biochemical systems and the generation of biochemical networks. This is illustrated on a fragment of the epidermal growth factor receptor (EGFR) signaling cascade and currently implemented in *Tom*.

6.2.6. Rewriting-Based Access Control Policies

Participants: Yassine Guebbas, Claude Kirchner, Hélène Kirchner, Anderson Santana, Eric Ke Wang.

Security policies, in particular access control, are fundamental elements of computer security. In collaboration with Dan Dougherty (Worcester Polytechnic Institute, USA) we have addressed the problem of authoring and analyzing policies in a modular way using techniques developed in the field of term rewriting, focusing especially on the use of rewriting strategies. Term rewriting supports a formalization of access control with a clear declarative semantics based on equational logic and an operational semantics guided by strategies. Well-established term rewriting techniques allow us to check properties of policies such as the absence of conflicts and the property of always returning a decision. A rich language for expressing rewriting strategies is used to define a theory of modular construction of policies, in which we can better understand the preservation of properties of policies under composition.

This framework is presented in [35] where the robustness of the approach is illustrated on the composition operators of XACML.

Despite the existence of a vast literature on access control, it is still very hard to assure the compliance of a large system to a given dynamic access control policy. Based on the formal islands approach, we provided in [44] a systematic methodology to weave dynamic, formally specified policies on existing applications using aspect-oriented programming. To that end, access control policies are formalized using term rewriting systems,

allowing us to have an agile, modular, and precise way to specify and to ensure their formal properties. These high-level descriptions are then weaved into the existing code, such that the resulting program implements a safe reference monitor for the specified policy. For developers, this provides a systematic process to enforce dynamic policies in a modular and flexible way. The level of reuse is improved because policies are independently specified and checked, to be later weaved into various different applications. We implemented the approach using *Tom*, and *AspectJ*. Test cases gave quite encouraging results.

Another contribution in this domain, in collaboration with Charles Morisset (LIP6), has been to address the verification of information leakage. Although this important property is assumed in several access control models and is well-understood formally, it is hard to verify in actual implementations of a given security policy. In [45], we proposed a general algorithm that allows one to automatically identify information leakage by model-checking rewrite-based access control policies. This approach is illustrated on the well-known model of Bell and LaPadula for multi-level policies, and we showed that its generalization, as proposed by McLean, does not respect the property.

6.2.7. *Quotient types in functional programming*

Participants: Frédéric Blanqui, Richard Bonichon, Laura Lowenthal.

This work was done in collaboration with Thérèse Hardin (LIP6) and Pierre Weis (INRIA Paris - Rocquencourt).

Many algorithms use concrete data types with some additional invariants. The set of values satisfying the invariants is often a set of representatives for the equivalence classes of some equational theory. For instance, a sorted list is a particular representative wrt commutativity. Theories like associativity, neutral element, idempotence, etc. are also very common. Now, when one wants to combine various invariants, it may be difficult to find the suitable representatives and to efficiently implement the invariants. The preservation of invariants throughout the whole program is even more difficult and error prone. Classically, the programmer solves this problem using a combination of two techniques: the definition of appropriate construction functions for the representatives and the consistent usage of these functions ensured via compiler verifications. The common way of ensuring consistency is to use an abstract data type for the representatives; unfortunately, pattern matching on representatives is lost. A more appealing alternative is to define a concrete data type with private constructors so that both compiler verification and pattern matching on representatives are granted. In [24], we detailed the notion of private data type and studied the existence of construction functions. We also described a prototype, called Moca (see Section 5.4), that addresses the entire problem of defining concrete data types with invariants: it generates efficient construction functions for the combination of common invariants and builds representatives that belong to a concrete data type with private constructors.

6.2.8. *Polygraphs as a computational model*

Participants: Yves Guiraud, Aurélien Monot, Pierre-Etienne Moreau.

Polygraphs provide an algebraic structure to graphical computations. Introduced by Albert Burroni as a n-categorical formalization of equational theories [58], they also describe in a uniform way several kinds of objects, coming from different fields of science: abstract, word and term rewriting systems [75], [68]; Petri nets [70]; propositional classical and linear logics [69]; abstract algebraic structures [77]; braids, knots and tangle diagrams with Reidemeister moves [67]; Feynman and Penrose diagrams [52].

With Guillaume Bonfante (Carte, LORIA and INRIA), we have defined polygraphic programs as a generalisation of first-order functional programs. Inspired by polynomial interpretations of terms [56], we have built complexity analysis tools, called polygraphic interpretations, relying on the structure of n-category. These tools allowed us to give a new, polygraphic characterization of the complexity class of functions that are computable in polynomial time [27], [12]. Moreover, polygraphic interpretations can prove termination of existing first-order functional programs [51].

To explore theoretical aspects, we work with François Lamarche (Calligramme, LORIA and INRIA) on foundations and with Philippe Malbos (Institut Camille Jordan, Lyon) on homological tools for polygraph analysis. On the practical side, we have started the development of Cat, an environment for certified polygraphic programming: as a first step, a compiler of polygraphic programs in the *Tom* language is currently developped.

6.3. Mechanized deduction

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

On one hand, we have obtained new or refined results for proving properties of probabilistic, rule-based and functional programs, in particular termination, using both the inductive and size-based approaches. We also began to develop a *Coq* library for certifying termination proofs.

On the other hand, we have further studied deduction modulo, identified an intrinsic complexity measure of a proof and a notion of "good proofs" in automated deduction, and explored the new concept of superdeduction.

6.3.1. Rewriting and probabilities

Participants: Florent Garnier, Claude Kirchner.

Florent Garnier has defended his PhD thesis "Terminaison en temps moyen fini de système de règles probabilistes", supervised by Claude Kirchner and Olivier Bournez (Carte team at LORIA), in September 2007 [10]. It includes earlier results published in 2005 and 2006 in the RTA conference, as well as unpublished ones. They deal with a refined way to study the termination in a finite mean time under strategies. This work also introduced a formalism to synchronize timed automaton that share a common communication medium, in order to simulate a pool of computers exchanging messages through a common radio channel. The latter formalism is used to model a pool of computers communicating using the CSMA/CA protocol and to prove that this protocol can terminate within a finite mean time when all stations start simultaneously. Finally, probabilistic rewrite systems are encoded in the *Tom* language.

6.3.2. Proving Properties of Reduction Relations

Participant: Hélène Kirchner.

In collaboration with Isabelle Gnaedig (Carte team at LORIA), we described in [37] an inductive proof method for properties of reduction relations, inspired from our previous work on proving, by explicit induction, termination of rewriting under strategies [14]. The reduction trees are simulated with proof trees generated by narrowing and an abstraction mechanism. While narrowing simulates reduction, abstraction relies on the induction principle to replace subterms by variables representing specific reduced forms that trivially satisfy the property to be proved. The induction ordering is not given a priori, but defined with ordering constraints, incrementally set during the proof. Abstraction constraints are used to control the narrowing mechanism, well-known to easily diverge. The proof method is briefly illustrated on various examples of properties: (weak-)termination under a strategy, definition completeness and existence of constructor forms, (weak-)reducibility of requests to specific answers, termination of probabilistic rewriting, termination of a transition system.

6.3.3. Strong Normalization with Union and Existential Types

Participant: Colin Riba.

In [55], Frédéric Blanqui and Colin Riba have proposed a termination criterion for higher-order conditional rewriting that use constrained types. Existential constraints arise for example when proving that QuickSort preserves the size of its argument. The criterion thus relies on proof methods for the strong normalization of typed λ-calculus plus rewriting in presence of implicit existential types. Usually, such proofs use interpretations of types by sets of strongly normalizing terms and rely on the soundess of the interpretations: typable terms belongs to the interpretation of their types. Soundness requires types to be interpreted by sets of terms satisfying some closure conditions, while union and implicit existential types are naturally interpreted by unions of type interpretations. However, in presence of rewriting, it is not trivial to find sound closure conditions that are preserved by union. We have made two contributions to this question.

First, we have given a necessary and sufficient condition for the closure condition of Girard's reducibility candidates to be stable by union [42]. Our condition is that Girard's candidates are exactly the non-empty sets of strongly normalizing terms that are downward closed for a weak observational preorder. Moreover, we have shown that this condition is met for the pure λ-calculus as well as for the λ-calculus enriched with product, co-product and recursive types.

Second, in [43] we questioned the possibility of having, for a given rewrite system, a sound type interpretation which is stable by union. Using a type system featuring union types, we have shown that there exists a confluent rewrite system that does not admit a sound type interpretation stable by union. The problem comes from the elimination rule of union types, which may allow typing non strongly normalizing terms. Moreover, we have studied sound interpretations of union types that are not stable by union. We have shown that a type interpretation based on biorthogonality is maximal among such interpretations: for a class of simple rewrite systems, this type interpretation is sound if and only if the terms typable with the elimination rule of union types are strongly normalizing.

6.3.4. Building Decision Procedures in the Calculus of Inductive Constructions

Participant: Frédéric Blanqui.

This work was done in collaboration with Jean-Pierre Jouannaud and Pierre-Yves Strub (LIX, Ecole Polytechnique).

It is commonly agreed that the success of future proof assistants will rely on their ability to incorporate computations within deductions in order to mimic the mathematician when replacing the proof of a proposition P by the proof of an equivalent proposition P' obtained from P thanks to possibly complex calculations.

In [26], we have investigated a new version of the calculus of constructions which incorporates arbitrary decision procedures into deductions via the conversion rule of the calculus. Besides the novelty of the problem itself in the context of the calculus of constructions, a major technical innovation of this work lies in the fact that the computation mechanism varies along proof-checking: goals are sent to the decision procedure together with the set of user hypotheses available from the current context.

Our main result shows that this extension of the calculus of constructions does not compromise its main properties: confluence, strong normalization and decidability of proof-checking are all preserved. We also showed in detail how a goal to be proved in the calculus of constructions is actually transformed into a goal in a decidable first-order theory. Based on this transformation, we are currently developing a new version of Coq implementing this calculus, taking linear arithmetic and the theory of lists as targets combined via Shostak's algorithm.

6.3.5. Termination of higher-order rewrite systems

Participant: Frédéric Blanqui.

This work was done in collaboration with Jean-Pierre Jouannaud (LIX, Ecole Polytechnique) and Albert Rubio (Technical University of Catalonia).

In [23], we surveyed the notion of computability closure and proved new results about computability and higher-order matching. The notion of computability closure has been introduced for proving the termination of higher-order rewriting with first-order matching by Jean-Pierre Jouannaud and Mitsuhiro Okada in a 1997 draft which later served as a basis for F. Blanqui's PhD. In this paper, we showed how this notion can also be used for dealing with beta-normalized rewriting with matching modulo beta-eta (on patterns à la Miller), rewriting with matching modulo some equational theory, and higher-order data types (types with constructors having functional recursive arguments). Finally, we showed how the computability closure can easily be turned into a reduction ordering which, in the higher-order case, contains Jean-Pierre Jouannaud and Albert Rubio's higher-order recursive path ordering and, in the first-order case, is equal to the usual first-order recursive path ordering.

In [25], we provided a new, decidable definition of the higher-order recursive path ordering in which type comparisons are made only when needed, therefore eliminating the need for the computability closure, and bound variables are handled explicitly, making it possible to handle recursors for arbitrary strictly positive inductive types.

6.3.6. Simple proofs in deduction modulo

Participants: Guillaume Burel, Claude Kirchner.

Solving goals, like deciding word problems or resolving constraints, is much easier in some theory presentations than in others. We have designed a general proof-theoretic framework centered around well-founded orderings of proofs and within which completion-like processes can be modeled around notions of saturation and redundancy [4].

In [31], [46], using this framework, we designed a new completion procedure which permits to regain the cut admissibility in deduction modulo [5], which does not hold in general if propositions can be rewritten [71]. We first proved that this property is in fact undecidable. Then, we proved that deduction modulo fits in the general framework, in such a way that the better presentations admit cuts. The framework gives therefore a completion procedure that permits to recover the cut admissibility. Based on a tableau method for deduction modulo, it is actually implemented in *Tom*.

We also focused on the length of proofs in deduction modulo. Parikh proved a speed-up theorem stated by Gödel: proofs in second-order arithmetic can be unboundedly shorter than in first order [81]. In [30] we defined a translation of second-order arithmetic into first-order arithmetic *modulo* which is linear, i.e. the length of proofs remains the same. The speed-up lies therefore in first-order arithmetic, depending whether we work modulo or not. This result allowed us to prove that the speed-up can be expressed as simple computation, therefore justifying the use of deduction modulo as an efficient first-order setting simulating higher order.

6.3.7. *Superdeduction*

Participants: Horatiu Cirstea, Paul Brauner, Jonathan Demange, Clément Houtman, Claude Kirchner.

Following the seminal work of Benjamin Wack on extended natural deduction [83], we introduced superdeduction [28] which is a new systematic way of extending deduction systems with rules derived from an axiomatic theory. Since it explicitly deals with *deduction*, we presented it as a complementary approach to *deduction modulo* [5] which deals with *computation*. The superdeduction extension proposes to infer new deduction rules from part of the theory in a sound, systematic and complete way. First, we presented its application to classical sequent calculus and we proved its soundness and completeness. After exhibiting a proof-term language associated with the corresponding deduction system, we proved its strong normalisation under non-trivial hypothesis, therefore ensuring the consistency of instances of the system, as well as of a large class of theories.

The proof of strong normalization can be found in [29], as well as significant examples including higher-order logic, induction and equality explaining why superdeduction could be a grounding framework for a new generation of interactive proof environments.

Finally, we pointed out the benefits of superdeduction in the frame of interactive proof building by developing an implementation of superdeduction modulo using the *Tom* language. This prototype called *Lemuridæ* can be downloaded from *Tom*'s CVS. The theoretical foundations of such a framework have been studied in [50], which relates the strong normalization property of superdeduction to the one of deduction modulo.

6.3.8. *Inductive proof search*

Participants: Claude Kirchner, Hélène Kirchner, Fabrice Nahon.

In the line of previous work on a proof theoretic framework to perform rewrite based inductive reasonning, Fabrice Nahon's PhD thesis [11] proposed an original narrowing-based proof search method for inductive theorems. It has the specificity to be grounded on deduction modulo and to rely on narrowing to provide both induction variables and instantiation schemes. It also yields a direct translation from a successful proof search derivation to a proof in the sequent calculus. The method is shown to be sound and refutationally correct in a proof theoretical way. The first approach presented in [63], [73] has been extended to equational rewrite theories given by a rewrite system $\mathcal{R}$ and a set E of equalities [41].

Whenever the equational rewrite system $(\mathcal{R}, E)$ has good properties of termination, sufficient completeness, and whenever E is constructor preserving, narrowing at defined-innermost positions is performed with unifiers which are constructor substitutions. This is especially interesting for associative and associative-commutative theories for which the general proof search system is refined.

7. Other Grants and Activities

7.1. Glossary

ANR National Agency for Research
ARC
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

7.2. National initiatives

We participate in the "Logic and Complexity" part of the GDR–IM (CNRS Research Group on Mathematical Computer Science), in the projects "Logic, Algebra and Computation" (mixing algebraic and logical systems) and "Geometry of Computation" (using geometrical and topological methods in computer science).

7.2.1. ARC Quotient 2007-2008

Participants: Frédéric Blanqui, Richard Bonichon, Laura Lowenthal.

This project gathers people from INRIA Nancy - Grand Est (Frédéric Blanqui, Richard Bonichon, Laura Lowenthal), INRIA Paris - Rocquencourt (Pierre Weis and Damien Doligez), Université Paris 6 (Thérèse Hardin, Renaud Rioboo) and CNAM (David Delahaye, Catherine Dubois). Its aim is to study and certify the use of non-free concrete data types in functional programming (see Section 6.2.7), and develop an extension of OCaml providing such types (see Section 5.4).

7.2.2. Infer

Participants: Guillaume Burel, Claude Kirchner.

This ANR project is a grouping of three teams through their common interest for a new approach to proof theory, called "deep inference". The project aims at refining its potential and at applying it to problems related to the foundations of logic and to more practical questions in the algorithmic of deductive systems, such as identity of proofs, Curry-Howard isomorphism, complexity of proofs, formulation of "exotic" logical systems, links with other paradigms like deduction modulo, etc. For more information, see the Infer website at *http://www.lix.polytechnique.fr/~lutz/orgs/infer.html*.

7.2.3. Inval

Participant: Yves Guiraud.

The ANR project "Invariants algébriques des systèmes informatiques" (Inval), headed by Éric Goubault (CEA Saclay), federates researchers in mathematics and theoretical computer science. Its main objective is to favour the transfer of ideas and methods between both communities. The coordinator for the LORIA site is François Lamarche (Calligramme). An Inval meeting was held at LORIA on September 7, 2007. For more information, see the Inval website at *http://www.pps.jussieu.fr/~inval/index.html*.

7.2.4. Ravaj

Participants: Émilie Balland, Yohan Boichut, Martin Grandcolas, Pierre-Etienne Moreau, Baptiste Payan.

Ravaj (Réécriture et Approximation pour la Vérification d'Applications Java) is an ANR project coordinated by Thomas Genet (Irisa). The goal is to model Java bytecode programs using term rewriting and to use completion techniques to compute the set of reachable terms. Then, it is possible to check some properties related to reachability (in particular safety and security properties) on the modeled system using tree automata intersection algorithms.

7.2.5. SSURF

Participants: Horatiu Cirstea, Anderson Santana.

"SSURF: Safety and Security under FOCAL" is an ANR project coordinated by Mathieu Jaume (LIP6). The SSURF project consists in characterizing and studying the required features that an Integrated Development Environment (IDE) must provide in order not only to obtain software systems in conformance with high Evaluation Assurance Levels (EAL-5, 6 and 7), but also to ease the evaluation process according to various standards (e.g. IEC61508, CC, ...). Moreover we aim at developing a formal generic framework describing various security properties, e.g. access control policies, together with their implementations using such an IDE.

7.3. International networks and working groups

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.

7.4. 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.
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.

7.5. Visiting scientists

- Anamaria Martins-Moreira, Brazil, two weeks, January–February 2007.
- Andreas Abel, Munich, one week, February–March 2007.
- Philippe Malbos, Lyon, one week, September 2007.

7.6. Invited lecturers

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

- Quang-Huy Nguyen (Security Labs, Gemalto), *Security evaluation and formal verification*.
- Serge Autexier (Saarland University), *Integrating the text-editor TeXmacs with the proof assistance system Omega using Plato*.
- François-Régis Sinot (Univesidade do Porto), *More laziness!*
- Clara Bertolissi (Université Aix-Marseille 1), *The rewriting calculus as a combinatory reduction system*.
- Andreas Abel (Ludwig-Maximilians-Universität München), *Normalization by evaluation and dependent types*.
- Manuel Maarek (Heriot-Watt University Edimburgh), *Restoring natural language as a computerised mathematics input method*.
- Mathieu Jaume (Université Paris 6), *Comparaison de politiques d'accès*.
- Makoto Tatsuta (National Institute for Informatics Tokyo), *Simple saturated sets for disjunction and second-order existential quantification*.
- Arnaud Bailly (Oqube), *Une étude de cas sur l'adaptation et le développement rapide d'applications de vente en ligne multicanaux*.

8. Dissemination

8.1. Leadership within scientific community

AFIT French chapter of EATCS

ASIAN Asian Computing Science Conference

CISSI Comité Interministériel sur la Sécurité des Systèmes d'Information

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

STACS Symposium on Theoretical Aspects of Computer Science

- Frédéric Blanqui:
  - Program committee of the 1st international workshop on type theory, proof theory, and rewriting (TPR'07), 29 June 2007, Paris, in conjonction with RDP'07. See http://www.lix.polytechnique.fr/~dowek/tpr.html.
  - Organizer of the 1st internation workshop on the certification of termination proofs, Nancy, 11-12 May 2007. See http://color.loria.fr/.
- Horatiu Cirstea:
  - Program committees of RULE 2007, WRS 2007.
- Yves Guiraud:
  - Organization committee of the 4th Inval meeting (Nancy, September 7).
  - Organization committee of the 86th Peripathetic Seminar on Sheaves and Logic (Nancy, September 8-9).
- Claude Kirchner:
  - Since June, delegate director of the FUTURS INRIA research center, Bordeaux site.
  - Chair of the scientific committee for the national ACISI programs.
  - Chair of the evaluation committee of the ANR SESUR2007 program.
  - Chair until June of the LORIA building extension committee.
  - Co-coordinator of the Franco-Japanese 3 years cooperation program on security founded by CNRS and JST.
  - Editorial boards of *Journal of Automated Reasoning*, *Journal of Applied Logic*.
  - Program committee of LSFA'07, Brazilian Workshop on Logical and Semantic Frameworks, with Applications, Chair of the scientific committee of the second international school on Rewriting (ISR'2007).
  - Chair of the IFIP WG 1.6 working group on rewriting and applications.
  - Member of the advisory board of LICS.
  - Member of the working group on research and perspectives of the CISSI.
  - Co-organizer of the Symposium in Honor of Jean-Pierre Jouannaud's 60th birthday (Cachan, june 21-22, 2007)
- Hélène Kirchner:
  - Director of LORIA and INRIA Lorraine until January 31, 2007. Then Deputy scientific director at INRIA.
  - 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 RTA'07 and LPAR'07.
  - Co-organizer of the Symposium in Honor of Jean-Pierre Jouannaud's 60th birthday (Cachan, june 21-22, 2007)
  - Member of Scientific directorate of the Dagstuhl international conference center.

  - Member of the ANR selection committee of the programmes “Non thématique 2007” and “Jeunes chercheurs 2007” in “Sciences et Technologies de l’Information”.
  - Chair of the evaluation board of LINA (Laboratoire d’Informatique de Nantes).
  - Member of the Conseil de Surveillance du GIS (Groupement d’Intérêt Scientifique) S3GS (Surveillance, sûreté et sécurité des grands systèmes).

- Pierre-Etienne Moreau:
  - Program committee of LSFA 2007 Brazilian Workshop on Logical and Semantic Frameworks, with Applications, and RULE 2007 International Workshop on Rule-Based Programming.
  - Chair with Sandrine Blazy of JFLA’07 Journées Francophones des Langages Applicatifs.
  - Steering committee of LDTA Workshop on Language Descriptions, Tools and Applications.

8.2. 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.

- Horatiu Cirstea:
  - Master course in Nancy on programming and proving with rule based languages, with Claude Kirchner and Pierre-Etienne Moreau.
  - Course on rewriting techniques and transformation at JFLA 2007 (Journées Francophones des Langages Applicatifs), with Antoine Reilles.

- Claude Kirchner:
  - Master course in Nancy on programming and proving with rule based languages, with Horatiu Cirstea and Pierre-Etienne Moreau.

- Pierre-Etienne Moreau:
  - Master course in Nancy on programming and proving with rule based languages, with Horatiu Cirstea and Claude Kirchner.
  - Lectures at ESIAL on fundamental data-structures.
  - Lecture at ISR (International School on Rewriting).

8.3. Invited talks

- Frédéric Blanqui:
  - Invited talk on the “Automated certification of termination proofs” at TYPES’07, 2 May 2007, Cividale del Friuli (Udine), Italy.

- Yves Guiraud:
  - Université Paris 7, “Polygraphic programs”, April 2, 2007.

- Claude Kirchner:
  - Colloque en l’honneur de Jean-Pierre Jouannaud: *Superdeduction at work*;
  - Dixième anniversaire du LSV: *Security challenges for computer systems*;
  - Colloque final du projet Asphales: *Les actions de recherche en sécurité*;
  - Canada-France meeting on security: *Security Research in France* and *Tools for specifying and verifying software*.

- Pierre-Etienne Moreau:
  - "Implementing Program Transformation with Tom and Java", GTTSE.
  - "Rules and Strategies in Java", WRS.

8.4. Visits

- Yves Guiraud:
  - One week at Institut Camille Jordan, Lyon, June 2007.
- Hélène Kirchner:
  - In June 4-8, 2007, in Tokyo, visits of AIST (Research Center for Information Security), Hitachi (Systems Development Laboratory), Sony (Computer Science Laboratory), Keio University (Keio Research Institute for Digital Media and Content) and NII.

8.5. Thesis and admission committees

- Frédéric Blanqui:
  - Substitute member of the Saint-Etienne University recruitment committee (section 27).
  - Member of the PhD committee of Colin Riba on "Définitions par réécriture dans le lambda-calcul: confluence, réductibilité et typage".
- Horatiu Cirstea:
  - Germain Faure "Structures et modèles de calculs de réécriture", PhD (co-advisor)
  - Member of recruitment committee (section 27) of Nancy2.
- Claude Kirchner:
  - Eric Filiol "Modèles Booléens en virologie et en cryptologie", HDR (referee);
  - Yann Radenac "Programmation "chimique" d'ordre supérieur", PhD (referee);
  - Thierry Sans "Beyond Access Control - Specifying and Deploying Security Policies in Information Systems", PhD;
  - Germain Faure "Structures et modèles de calculs de réécriture", PhD (co-advisor);
  - Luigi Liquori "Peter, the Language that does not Exist ... ", HDR;
  - Florent Garnier "Terminaison en temps moyen fini de systèmes de règles probabilistes", PhD (co-advisor);
  - Charles Morisset "Définition d'un cadre sémantique pour la spécification, l'implantation et la comparaison de modèles de contrôle d'accès", PhD;
  - Loic Duflot "Contribution à la sécurité des systèmes d'exploitation et des microprocesseurs", PhD (referee);
  - Fabrice Nahon "Preuve par récurrence dans le calcul des séquents modulo", PhD (co-advisor);
  - Romain Péchoux "Analyse de la complexité des programmes par interprétation sémantique" PhD;
  - Colin Riba "Définitions par réécriture dans le lambda-calcul : confluence, réductibilité et typage" PhD (co-advisor).
- Hélène Kirchner:
  - Duc Khan-Tran "Conception de procédures de décision par combinaison et saturation", PhD (co-advisor)

  - Fabrice Nahon “Preuve par récurrence dans le calcul des séquents modulo”, PhD (co-advisor)
  - Member of recruitment committees (section 27) of UHP Nancy1, Nancy2, INPL.
- Pierre-Etienne Moreau:
  - Member of the UHP recruitment committee (section 27).

9. 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] 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.

[4] N. DERSHOWITZ, C. KIRCHNER. *Abstract canonical presentations*, in "Theoretical Computer Science", vol. 357, n^{o} 1-3, July 2006, p. 53–69.

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

[6] 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.

[7] 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.

[8] 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

[9] G. FAURE. *Structures et modèles de calculs de réécriture*, Ph. D. Thesis, Université Henri Poincaré - Nancy 1, jul 2007, http://tel.archives-ouvertes.fr/tel-00164576/en/.

[10] F. GARNIER. *Terminaison en temps moyen fini de systèmes de règles probabilistes*, Ph. D. Thesis, Institut National Polytechnique de Lorraine - INPL, sep 2007, http://tel.archives-ouvertes.fr/tel-00186774/en/.

[11] F. NAHON. *Preuves par induction dans le calcul des séquents modulo*, Ph. D. Thesis, Université Henri Poincaré – Nancy 1, October 2007.

Articles in refereed journals and book chapters

[12] G. BONFANTE, Y. GUIRAUD. *Polygraphic programs and polynomial-time functions*, in "Logical Methods in Computer Science (LMCS)", 38 pages, under revision, 2007, http://hal.inria.fr/inria-00122932/en/.

[13] H. CIRSTEA, G. FAURE, C. KIRCHNER. *A Rho-Calculus of explicit constraint application*, in "Higher-Order and Symbolic Computation", vol. 20, 2007, p. 37-72, http://hal.inria.fr/inria-00000628/en/.

[14] I. GNAEDIG, H. KIRCHNER. *Termination of Rewriting under Strategies*, in "ACM Transactions on Computational Logic", to appear, 2007, http://hal.inria.fr/inria-00182432/en/.

[15] B. WACK, C. HOUTMANN. *Strong Normalization in two Pure Pattern Type Systems*, in "Mathematical Structures in Computer Science", to appear, 2007, http://hal.inria.fr/inria-00186815/en/.

Publications in Conferences and Workshops

[16] O. ANDREI, H. KIRCHNER. *A Rewriting Calculus for Multigraphs with Ports*, in "The Eighth International Workshop on Rule-Based Programming - RULE 2007), Paris France", Electronic Notes in Theoretical Computer Science, 2007, 20, http://hal.inria.fr/inria-00139363/en/.

[17] O. ANDREI, H. KIRCHNER. *Graph Rewriting and Strategies for Modeling Biochemical Networks*, in "International Workshop on Natural Computing and Applications - NCA 2007, Timisoara, Roumanie", IEEE Computer Society, 2007, http://hal.inria.fr/inria-00146362/en/.

[18] P. BALDAN, C. BERTOLISSI, H. CIRSTEA, C. KIRCHNER. *Towards a sharing strategy for the graph rewriting calculus*, in "7th International Workshop on Reduction Strategies in Rewriting and Programming - WRS 2007, Paris, France", J. GIESL (editor), Electronic Notes in Theoretical Computer Science, Elsevier, 2007, http://hal.inria.fr/inria-00186141/en/.

[19] E. BALLAND, P. BRAUNER. *Term-graph rewriting in Tom using relative positions*, in "4th International Workshop on Computing with Terms and Graphs - TERMGRAPH 2007, Braga, Portugal", Electronic Notes in Theoretical Computer Science, mar 2007, http://hal.inria.fr/inria-00129515/en/.

[20] E. BALLAND, P. BRAUNER, R. KOPETZ, P.-E. MOREAU, A. REILLES. *Tom: Piggybacking rewriting on java*, in "Conference on Rewriting Techniques and Applications - RTA'07, Paris, France", Lecture Notes in Computer Science, jun 2007, http://hal.inria.fr/inria-00142045/en/.

[21] E. BALLAND, P.-E. MOREAU, A. REILLES. *Bytecode rewriting in Tom*, in "Second Workshop on Bytecode Semantics, Verification, Analysis and Transformation - BYTECODE 07, Braga, Portugal", mar 2007, http://hal.inria.fr/inria-00129513/en/.

[22] C. BERTOLISSI, C. KIRCHNER. *The Rewriting Calculus as a Combinatory Reduction System*, in "Tenth International Conference on Foundations of Software Science and Computations Structures - FoSSaCS 2007, Braga, Portugal", H. SEIDL (editor), Lecture Notes in Computer Science, Springer-Verlag, mar 2007, http://hal.inria.fr/inria-00121792/en/.

[23] F. BLANQUI. *Computability Closure: Ten Years Later*, in "Colloquium in honor of Jean-Pierre Jouannaud, Cachan, France", Lecture Notes in Computer Science, vol. 4600, 2007, http://hal.inria.fr/inria-00161092/en/.

[24] F. BLANQUI, T. HARDIN, P. WEIS. *On the implementation of construction functions for non-free concrete data types*, in "16th European Symposium on Programming - ESOP'07, Braga, Portugal", Lecture Notes in Computer Science, vol. 4421, 2007, http://hal.inria.fr/inria-00095110/en/.

[25] F. BLANQUI, J.-P. JOUANNAUD, A. RUBIO. *HORPO with Computability Closure : A Reconstruction*, in "14th International Conference on Logic for Programming Artificial Intelligence and Reasoning, Yerevan, Arménie", Lecture Notes in Computer Science, vol. 4790, oct 2007, http://hal.inria.fr/inria-00168304/en/.

[26] F. BLANQUI, J.-P. JOUANNAUD, P.-Y. STRUB. *Building Decision Procedures in the Calculus of Inductive Constructions*, in "16th Annual Conference on Computer Science and Logic - CSL 2007, Lausanne, Suisse", J. DUPARC, T. HENZIGER (editors), Lecture Notes in Computer Science, vol. 4646, Springer Verlag, 2007, http://hal.inria.fr/inria-00160586/en/.

[27] G. BONFANTE, Y. GUIRAUD. *Intensional properties of polygraphs*, in "4th International Workshop on Computing with Terms and Graphs - TERMGRAPH 2007, Braga, Portugal", Electronic Notes in Theoretical Computer Science, 12 pages, to appear, 2007, http://hal.inria.fr/inria-00129391/en/.

[28] P. BRAUNER, C. HOUTMANN, C. KIRCHNER. *Principles of Superdeduction*, in "Twenty-Second Annual IEEE Symposium on Logic in Computer Science - LiCS 2007, Wroclaw, Pologne", IEEE Computer Society, 2007, http://hal.inria.fr/inria-00133557/en/.

[29] P. BRAUNER, C. HOUTMANN, C. KIRCHNER. *Superdeduction at work*, in "Colloquium in honor of Jean-Pierre Jouannaud - Rewriting, Computation and Proof Lecture Notes in Computer Science, Cachan, France", H. COMON-LUNDH, C. KIRCHNER, H. KIRCHNER (editors), Lecture Notes in Computer Science, vol. 4600, Springer, 2007, p. 132-166, http://hal.inria.fr/inria-00141672/en/.

[30] G. BUREL. *Unbounded Proof-Length Speed-up in Deduction Modulo*, in "16th Annual Conference on Computer Science and Logic - CSL 2007, Lausanne, Suisse", J. DUPARC, T. HENZIGER (editors), Lecture Notes in Computer Science, vol. 4646, Springer Verlag, 2007, p. 496-511, http://hal.inria.fr/inria-00138195/en/.

[31] G. BUREL, C. KIRCHNER. *Cut Elimination in Deduction Modulo by Abstract Completion*, in "Symposium on Logical Foundations of Computer Science - LFCS'07, New York, États-Unis d'Amérique", S. ARTEMOV, A. NERODE (editors), Lecture Notes in Computer Science, vol. 4514, Springer Verlag, 2007, p. 115-131, http://hal.inria.fr/inria-00115556/en/.

[32] H. CIRSTEA, G. FAURE. *Confluence of Pattern-Based Calculi*, in "18th International Conference on Term Rewriting and Applications - RTA 2007, Paris, France", F. BAADER (editor), Lecture Notes in Computer Science, vol. 4533, Springer Berlin / Heidelberg, 2007, p. 78-92, http://hal.inria.fr/inria-00185882/en/.

[33] 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 - WRS 2006, Seattle, États-Unis d'Amérique", S. ANTOY (editor), Electronic Notes in Theoretical Computer Science, vol. 174, n^{o} 10, Elsevier, 2007, p. 39-56, http://hal.inria.fr/inria-00000821/en/.

[34] H. CIRSTEA, C. HOUTMANN, B. WACK. *Distributive rewriting calculus*, in "Sixth International Workshop on Reduction Strategies in Rewriting and Programming, Vienne, Autriche", G. DENKER, C. L. TALCOTT (editors), Electronic Notes in Theoretical Computer Science, vol. 176, n^{o} 4, Elsevier, 2007, p. 95-111, http://hal.inria.fr/inria-00110867/en/.

[35] D. J. DOUGHERTY, C. KIRCHNER, H. KIRCHNER, A. SANTANA DE OLIVEIRA. *Modular Access Control via Strategic Rewriting*, in "12th European Symposium On Research In Computer Security - ESORICS 2007, Dresden, Allemagne", J. BISKUP, J. LOPEZ (editors), Lecture Notes in Computer Science, vol. 4734, Springer Berlin / Heidelberg, 2007, p. 578-593, http://hal.inria.fr/inria-00185697/en/.

[36] G. FAURE. *Term collection in lambda/rho-calculi*, in "2nd International Workshop on Developments in Computational Models - DCM 2006, Venise, Italie", Electronic Notes in Theoritical Computer Science, vol. 171, n^{o} 3, 2007, p. 3-19, http://hal.inria.fr/inria-00102993/en/.

[37] I. GNAEDIG, H. KIRCHNER. *Narrowing, Abstraction and Constraints for Proving Properties of Reduction Relations*, in "Rewriting, Computation and Proof - Essays Dedicated to Jean-Pierre Jouannaud on the Occasion of his 60th Birthday, Paris, France", H. COMON-LUNDH, C. KIRCHNER, H. KIRCHNER (editors), Lecture Notes in Computer Science, vol. 4600, Springer, 2007, p. 44-67, http://hal.inria.fr/inria-00182434/en/.

[38] C. KIRCHNER, R. KOPETZ, P.-E. MOREAU. *Anti-Pattern Matching*, in "16th European Symposium on Programming - ESOP'07, Braga, Portugal", mar 2007, http://hal.inria.fr/inria-00129419/en/.

[39] C. KIRCHNER, R. KOPETZ, P.-E. MOREAU. *Anti-Pattern Matching Modulo*, in "21th International Workshop on Unification - UNIF'07, Paris, France", É. CONTEJEAN (editor), 2007, http://hal.inria.fr/inria-00176055/en/.

[40] P.-E. MOREAU, A. REILLES. *Rules and Strategies in Java*, in "7th International Workshop on Reduction Strategies in Rewriting and Programming - WRS 2007, Paris, France", 2007, http://hal.inria.fr/inria-00185698/en/.

[41] F. NAHON, C. KIRCHNER, H. KIRCHNER. *Inductive Proof Search Modulo*, in "6th International Workshop on First-Order Theorem Proving - FTP 2007, Liverpool, Royaume-Uni", S. RANISE (editor), vol. Technical report ULCS-07-018, Department of Computer Science, University of Liverpool, 2007, p. 4-19, http://hal.inria.fr/inria-00187458/en/.

[42] C. RIBA. *On the Stability by Union of Reducibility Candidates*, in "10th International Conference on Foundations of Software Science and Computational Structures - FoSSaCS 2007, Braga, Portugal", Springer Verlag, 2007, p. 317-331, http://hal.archives-ouvertes.fr/hal-00123116/en/.

[43] C. RIBA. *Strong Normalization as Safe Interaction*, in "Twenty-Second Annual IEEE Symposium on Logic in Computer Science - LiCS 2007, Wroclaw, Poland", jul 2007, http://hal.inria.fr/inria-00130456/en/.

[44] A. SANTANA DE OLIVEIRA, E. KE WANG, C. KIRCHNER, H. KIRCHNER. *Weaving Rewrite-Based Access Control Policies*, in "The 5th ACM Workshop on Formal Methods in Security Engineering - FMSE 2007, Alexandria, États-Unis d'Amérique", H. MANTEL, V. GLIGOR (editors), ACM, 2007, http://hal.inria.fr/inria-00185710/en/.

[45] A. SANTANA DE OLIVEIRA, C. MORISSET. *Automated Detection of Information Leakage in Access Control*, in "Second International Workshop on Security and Rewriting Techniques - SecReT 2007, Paris, France", M. NESI (editor), 2007, http://hal.inria.fr/inria-00185713/en/.

Internal Reports

[46] G. BUREL, C. KIRCHNER. *Cut Elimination in Deduction Modulo by Abstract Completion (Full Version)*, extension de http://hal.inria.fr/inria-00115556 contenant toutes les preuves., Technical report, 2007, http://hal.inria.fr/inria-00132964/en/.

[47] C. KIRCHNER, R. KOPETZ, P.-E. MOREAU. *Anti-Pattern Matching Modulo*, Technical report, nov 2007, http://hal.inria.fr/inria-00129421/en/.

Miscellaneous

[48] E. BALLAND, P. BRAUNER, R. KOPETZ, P.-E. MOREAU, A. REILLES. *The Tom Manual*, 126 pages, 2007, http://tom.loria.fr/soft/release-2.5/manual-2.5/index.html.

[49] E. BALLAND, C. KIRCHNER, P.-E. MOREAU. *Term-graph rewriting via explicit paths*, 2007, http://hal.inria.fr/inria-00173535/en/.

[50] P. BRAUNER, G. DOWEK, B. WACK. *Normalization in Supernatural deduction and in Deduction modulo*, 8 pages, 2007, http://hal.inria.fr/inria-00141720/en/.

[51] Y. GUIRAUD. *Polygraphs for termination of left-linear term rewriting systems*, 15 pages, 2007, http://hal.inria.fr/inria-00129392/en/.

References in notes

[52] J. BAEZ, A. LAUDA. *A history of n-categorical physics*, Draft version, 2006.

[53] G. BARTHE, H. CIRSTEA, C. KIRCHNER, L. LIQUORI. *Pure Patterns Type Systems*, in "Proc. of POPL", The ACM press, Jan 2003, p. 250–261.

[54] 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.

[55] F. BLANQUI, C. RIBA. *Combining Typing and Size Constraints for Checking the Termination of Higher-Order Conditional Rewrite Systems*, in "Proc. 13th LPAR Conf., Phnom Penh (Cambodia)", LNAI, vol. 4246, Springer-Verlag, 2006.

[56] G. BONFANTE, A. CICHON, J.-Y. MARION, H. TOUZET. *Algorithms with polynomial interpretation termination proofs*, in "Journal of Functional Programming", vol. 11, n^{o} 1, January 2001, p. 33-53.

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

[58] 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.

[59] 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.

[60] 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.

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

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

[63] 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.

[64] D. DOLIGEZ, J. GARRIGUE, X. LEROY, D. RÉMY, J. VOUILLON. *The Objective Caml system release 3.10, Documentation and user's manual*, INRIA, France, 2007, http://caml.inria.fr/.

[65] L. DUCAS. *Certification de preuves de terminaison basées sur la décomposition du graphe des paires de dépendance*, Rapport de stage de fin de licence, 2007, http://color.loria.fr/.

[66] 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.

[67] Y. GUIRAUD. *Présentations d'opérades et systèmes de réécriture*, Ph. D. Thesis, Université Montpellier 2, June 2004.

[68] 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.

[69] 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.

[70] Y. GUIRAUD. *Two polygraphic presentations of Petri nets*, in "Theoretical Computer Science", vol. 360, n^{o} 1-3, August 2006, p. 124-146.

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

[72] 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.

[73] 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.

[74] A. KOPROWSKI, H. ZANTEMA. *Certification of Matrix Interpretations in Coq*, in "Proceedings of the 9th International Workshop on Termination - WST07", 2007, p. 9-12.

[75] Y. LAFONT. *Towards an algebraic theory of boolean circuits*, in "Journal of Pure and Applied Algebra", vol. 184, n^{o} 2-3, 2003, p. 257-310.

[76] 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.

[77] J.-L. LODAY. *Generalized bialgebras and triples of operads*, Preprint, 2006.

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

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

[80] 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.

[81] R. J. PARIKH. *Some results on the length of proofs*, in "Transactions of the ACM", vol. 177, March 1973, p. 29-36.

[82] TERESE. , M. BEZEM, J. W. KLOP, R. DE VRIJER (editors)*Term Rewriting Systems*, Cambridge University Press, 2002.

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