INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Project-Team Protheo

Constraints, Mechanized Deduction and Proofs of Software Properties

Lorraine

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).*

Team Leader
Claude Kirchner [DR INRIA]

Administrative assistants
Chantal Llorens [CNRS]

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

Faculty members
Carlos Castro [MC UHP, part time]
Horatiu Cirstea [MC Nancy2]
Olivier Fissore [ATER 01/09/03-31/08/04]

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

Technical staff
Julien Guyon [INRIA fixed-term engineer until 31/10/04]
Laika Moussa [INRIA fixed-term engineer from 15/04/04]
Anne-Claire Lonchamp [INRIA fixed-term engineer from 15/12/04]

Ph. D. students
Clara Bertolissi [MENRT since 01/10/02]
Germain Faure [ENS Cachan 01/10/03-30/09/04, MENRT since 01/10/04]
Florent Garnier [INRIA since 01/11/03]
Emmanuel Hainry [ENS Lyon 01/10/03-31/08/04, MENRT since 01/09/04, with Calligramme]
Liliana Ibănescu [INPL 01/01/01-31/08/04, ATER since 01/09/04]
Paulin Jacobé de Naurois [MENRT since 01/09/01, with Calligramme and Hong-Kong University]
Michael Moossen [INRIA 01/02/04-31/10/04]
Fabrice Nahon [High school teacher, since 01/09/02]
Antoine Reilles [CNRS since 01/10/03]
Colin Riba [MENRT since 01/10/04]
Anderson Santana [Brazil since 01/10/04]
Duc Tran [MENRT since 01/10/03]
Benjamin Wack [MENRT since 01/09/02]

Visiting scientists
Anamaria Martins Moreira [Brazil, 14/01/04-13/02/04]
Gopalan Nadathur [Minneapolis, 01/10/03-31/03/04]
Kellil Rabah [Algeria, 01/04/04-30/09/04]

Internships
Grégory Andrien [ESIAL, 28/06/04-27/08/04, with Pierre-Etienne Moreau]
Guillaume Darmont [Master, 09/02/04-31/07/04, with Pierre-Etienne Moreau]

Romain Demangeon [ENS Lyon, 01/06/04-16/07/04, with Claude Kirchner]
Chandan Kumar Dubey [IIT, 15/05/04-15/07/04, with Claude Kirchner]
Pierre Henneger [ESIAL, 28/06/04-27/08/04, with Pierre-Etienne Moreau]
Sébastien Hinderer [Master, 09/02/04-31/07/04, with Frédéric Blanqui]
Stéphane Hrvatin [Master, 29/03/04-31/08/04, with Christophe Ringeissen]
Nghiem Long Phan [IFI, 01/05/04-30/10/04, with Pierre-Etienne Moreau]

2. Overall Objectives

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 themes:

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

The team develops 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*.

The rewriting techniques have been developed since the ’70s 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 [76].

The 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 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 ρ-calculus [57]. 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 shown its 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 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 [69]. 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 lies on inductive reasoning or equation handling, and uses operator properties like associativity or commutativity.

Rewriting, which allows 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 developing specific termination proof tools for rewriting strategies.

Constraints allow to postpone complex symbolic problem solving, in order they can be solved in an expedient way. They also allow 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 to obtain much simpler automated proofs. Implementing these ideas has indeed allowed W. McCune [59][72] to solve an open mathematical problem. Combining constraints and rewriting based simplifications induces complex problems, as theoretical, for completeness of strategies, as practical, for efficiency of implementation. We explore these techniques both with a theoretical point of view and an experimental one.

4. Application Domains

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

Our researches apply 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 an ease in expressivity of 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 leaded 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: Pierre-Etienne Moreau, Julien Guyon, Antoine Reilles.

Since 2002, we have developed a new system called *Tom*, presented in [75]. This system consists in 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 [70]. 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.

5.3. CARIBOO

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

Participants: Olivier Fissore, 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 this year, *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 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 prove 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.

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

5.4. ELAN

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

Participants: Horatiu Cirstea, Olivier Fissore, Florent Garnier, Claude Kirchner, Hélène Kirchner, Pierre-Etienne Moreau, Christophe Ringeissen.

The *ELAN* [53][34] 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 takes from functional programming the concept of abstract data types and the function evaluation principle based on rewriting. But rewriting is inherently non-deterministic since several rules can be applied at different positions in a term and, in *ELAN*, a computation may have several results. This aspect is taken into account through choice operations and a backtracking capability. One of the main originality of the language is to provide strategy constructors to specify whether a function call returns several, at-least one or only one result. This declarative handling of non-determinism is part of a strategy language allowing the programmer to specify the control on rules application. This is in contrast to many existing rewriting-based languages where the term reduction strategy is hard-wired and not accessible to the designer of an application. The strategy language offers primitives for sequential composition, iteration, deterministic and non-deterministic choices of elementary strategies that are labeled rules. From these primitives, more complex strategies can be expressed. In addition the user can introduce new strategy operators and define them by rewrite rules. Evaluation of strategy application is itself based on rewriting. So the simple and well-known paradigm of rewriting provides both the logical framework in which deduction systems can be expressed and combined, and the evaluation mechanism of the language.

ELAN is documented, maintained and available at http://elan.loria.fr.

6. New Results

6.1. Rewriting calculus

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

Starting from our previous works [54], we noticed that the tool necessary to control rewriting must be explicit and can be described quite naturally by using rewriting too. This observation brought us to a new calculus generalizing λ-calculus and rewriting which we called ρ-calculus [57].

We have proposed and studied different instances of this calculus and their corresponding type systems. We have shown the expressive power of this calculus by giving a simple and typed encoding of standard strategies used in first-order rewriting. A version of the calculus handling explicitly matching and substitution applications has been also proposed. The explicit matching constraints have been also extended to (a restricted class of) unification constraints allowing one to represent, for example, infinite structures.

6.1.1. Explicit ρ-calculus

Participants: Horatiu Cirstea, Germain Faure, Claude Kirchner.

Theoretical presentations of the ρ-calculus often treat matching constraint computations as an atomic operation (although matching constraints are explicitly expressed). Concrete implementations have to take a much more realistic view: computations needed to find the solutions of a matching equation (in some matching theories) can have a strong impact on the properties of the calculus. Moreover, the application of the obtained substitutions usually involves term traversals.

Following the works on explicit substitutions for λ-calculus, we propose, study and exemplify [28] a ρ-calculus with explicit constraint handling, including substitution applications. The approach is modular and can be extended to arbitrary matching theories. We show that the calculus is powerful enough to deal with errors. We also prove the confluence of the calculus and the termination of the explicit constraint handling part.

The explicit substitution application studied so far is not optimal since the possible complexity of term traversals is not taken into account. We have been working on an improved version of the calculus that offers support for the composition of substitutions. Moreover, 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 [29].

6.1.2. Graphs in the ρ-calculus

Participants: Clara Bertolissi, Horatiu Cirstea, Claude Kirchner.

Starting from the classical untyped ρ-calculus and in collaboration with Paolo Baldan from Venizia University, we have proposed an extension with an enhanced expressive power that deals not only with terms but also with graphs and graph transformations [20].

The syntax of the ρ-calculus is adapted in order to describe graphs and the related notions such as sharing and cycles. The evaluation rules are also adapted for this new syntax leading to a calculus that we call *graph rewriting calculus*. In this new formalism we can represent and manipulate elaborated objects like, for example, regular infinite entities.

The calculus over terms is naturally generalized by using unification constraints in addition to the standard ρ-calculus matching constraints. Graph transformations are performed by explicit application of rewrite rules as first class entities. This therefore provides us with the basics for a natural extension of an explicit substitution calculus to term graphs.

6.1.3. Typed programming with ρ-calculus

Participants: Horatiu Cirstea, Claude Kirchner, Luigi Liquori, Benjamin Wack.

We have extensively studied a first-order typed ρ-calculus *à la* Church called $\rho^{Fix}_{\rightarrow}$, whose type system is mainly inspired by simply typed λ-calculus. The type system enjoys subject reduction, type uniqueness and decidability of typing. Moreover, the type system of $\rho^{Fix}_{\rightarrow}$ allows one to type (object oriented flavored) fixpoints, leading to an expressive and safe calculus. In particular, using pattern matching, one can encode and type-check term rewrite systems in a natural and automatic way [29].

Early versions of the (untyped) ρ-calculus have already been used to describe the implicit (leftmost-innermost) and user defined strategies of *ELAN* [57] but some *ad hoc* operators were added to the basic

calculus for this. The $\rho_{\rightarrow}^{Fix}$ presented here refers essentially to a simpler formulation of previously introduced versions of the rewriting calculus [58] where no new constructions have to be defined for expressing strategies for quite a large class of rewriting systems. The price to pay for this simplicity is that the encoded rewriting systems we handle are not the most general ones but still the most used ones in practice.

More recently, we have extended this study for two versions of a second-order ρ-calculus [35], one with full type annotations and the other with polymorphic type inference. Both systems can be considered as minimal calculi for polymorphic rewriting-based programming languages. The ability to type-check term rewriting systems and strategies ensures a good trade-off between expressiveness and safety of programs: the type system of $\rho_{\rightarrow}^{Fix}$ is suitable for static analysis, *i.e.* it ensures that functions get arguments of the expected type. However, as a wanted feature it does not enforce termination of typed terms: we have shown the encoding of some interesting terms leading to infinite reductions by the use of pattern matching features of the calculus.

The consequence is that our typing discipline fits for a programming language since we are interested in type consistency and in recursive (potentially non-terminating) programs. Conversely, it is not adapted for defining a logical framework, since normalization is strongly linked to consistency, so it definitively differs from other type systems we have proposed for the ρ-calculus [50].

As we have mentioned above, the automatic encoding we have proposed can be done only for a restricted class of rewrite systems and only some classical reduction strategies (*e.g.* outermost) have been encoded. Therefore, we currently investigate the way this can be generalized to unrestricted rewrite systems and to a larger class of reduction strategies.

6.1.4. Typed ρ-calculi for logics

Participants: Horatiu Cirstea, Claude Kirchner, Luigi Liquori, Benjamin Wack.

We are also interested in more elaborated type systems and we have proposed a ρ-cube which generalizes Barendregt's λ-cube. First, with Gilles Barthe and Luigi Liquori [50], we have proposed and studied a framework called Pure Pattern Type Systems (P^2TS), an algebraic extension of Pure Type Systems. In P^2TS, rewrite rules can be considered as lambda terms with patterns, and the application of a rule is the application of λ-calculus, enhanced with matching capabilities. This context uniformly unifies higher-order functions and rewriting mechanisms, together with a powerful type system. Thus, it is particularly adapted for formalizing the foundations of programming (meta)languages and proof assistants. We have proved various standard properties such as subject reduction and confluence.

More recently, we have proved the strong normalization for well-typed terms [37]. The proof is based on an encoding of terms and types from P^2TS into the λ-calculus. For this, we rely on an original compilation of the pattern matching capability of P^2TS into the λ-calculus. We show afterwards how to translate types: the expressive power of System *F*ω is needed in order to fully reproduce the original typing judgments of P^2TS. We prove that the encoding is complete with respect to reductions and typing, and we conclude with the strong normalization of simply-typed P^2TS terms.

We consider this work as a contribution in the study of correspondences between rewriting and logics. In this context, we can associate rewriting-based programs to ρ-terms and we are interested in finding a suitable logic and a Curry-Howard isomorphism between typed ρ-terms and this logic.

6.1.5. Models of the ρ-calculus

Participant: Germain Faure.

Up to now, the ρ-calculus has been studied only from the perspective of its operational semantics. In collaboration with Alexandre Miquel, we have proposed a first attempt to a Scott-style semantics for the ρ-calculus, when restricted to ML-style pattern-matching.

We define a notion of ρ-model in the category of Scott-domains, and prove the soundness of the reduction rules w.r.t. this notion of ρ-model. We then show that any universal domain $D = (D \rightarrow D)$ with a top element (including Scott's historical D_∞) can be given the structure of a ρ-model by interpreting the structure construction of the ρ-calculus by the binary sup. From this, we deduce that the full ACI-theory of the ρ-calculus is conservative w.r.t. the $\beta\eta$-theory for normalizable λ-terms.

6.2. Rule based programming

Keywords: *abstract data-type*, *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. Compilation of pattern-matching

Participants: Julien Guyon, Pierre-Etienne Moreau, Christophe Ringeissen.

In collaboration with Marian Vittek, we have designed a new formalism, called *Tom*, which allows us to integrate pattern matching facilities into imperative languages such as C, Java or Caml. This new formalism has been presented in [75].

Tom does not really define a new language: it is rather a language extension which adds new matching primitives to an existing imperative language. From an implementation point of view, it is a compiler which accepts different native languages: C, Java, and Caml. The compilation process consists of translating new matching constructs into the underlying host language. An important feature of *Tom* is to support equational matching. In particular, list matching, also known as associative matching with neutral element. Recently, we have introduced an XML extension which allows us to manipulate XML documents and to retrieve information via associative matching.

When developing an application which manipulates tree-like data structures (such as symbolic computation, compilation, XML transformation, message exchange, etc.), the need for matching can rarely be avoided, simply because some information has to be retrieved. *Tom* is particularly well suited for describing and implementing various kinds of applications where extraction of information has to be performed.

6.2.2. Representation of abstract data-types

Participants: Pierre-Etienne Moreau, Antoine Reilles.

In collaboration with Olivier Zendra from the DESIGN team, we have studied the garbage collector algorithm provided by the standard C implementation of the ATerm Library and stressed the fact that some peculiarities of this functional library could be taken advantage of by the memory management system. We have designed and implemented GC^2 [18], a new mark-and-sweep generational garbage collector for the ATerm Library that builds upon these peculiarities. This implementation improves the efficiency of all programs that use the ATerm Library. The source code is integrated into the main CVS Tree and is used to build a large number of tools.

In collaboration with Mark van den Brand and Jurgen Vinju of CWI at Amsterdam, we have presented a Java back-end for ApiGen [19], a new tool that generates implementations of abstract syntax trees. The generated code is characterized by strong typing combined with a generic interface and maximal sub-term sharing for memory efficiency and fast equality checking. The goal of this tool is to obtain safe and efficient programming interfaces for abstract syntax trees.

The contribution of this work is the combination of generating a strongly typed data-structure with maximal sub-term sharing in Java. Practical experience shows that this approach can not only be used for tools that are otherwise manually constructed, but also for internal data-structures in generated tools.

6.2.3. Environment for rule based programming

Participants: Gregory Andrien, Julien Guyon, Pierre Henneger, Stéphane Hrvatin, Pierre-Etienne Moreau, Antoine Reilles.

Based on the Eclipse Platform (an Integrated Development Environment), we have developed a new plugin for *Tom* [32]. This tool provides a specialized editor, an automatic build process, and an error management mechanism. By integrating an algebraic programming environment into Eclipse, we contribute to the promotion of formal methods and Eclipse to the educational, algebraic, and industrial communities.

In parallel to this approach, we have studied the design of a library which allows the definition of traversal strategies. Based on Stratego and the ρ-calculus, this library introduced a reduced set of basic strategies which can be combined, via a recursion operator μ. This results in a very flexible, efficient and easy to use library written in *Tom* and Java.

A new parser of *Tom* has been reimplemented during the internships of Pierre Henneger and Stéphane Hrvatin.

6.2.4. Program analysis and certification of program transformations

Participants: Claude Kirchner, Pierre-Etienne Moreau, Antoine Reilles.

We are studying the certification of the transformation of patterns from the *Tom* language into an imperative target language, like Java. By giving a certification of the fact that the *Tom* compiler produces correct target code for the *Tom* specification, we improve our confidence in the *Tom* compiler and the rule based approach. By giving, for each execution of the compiler a certification that the produced code meets the specification (or a failure), we can have much more information for debugging the compiler. It is particularly important in the case of associative matching for example.

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 to have 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. Then, using a just enough powerful intermediate language, which could be viewed as a subset of $C \cap Java \cap ML$, we formalize the behavior of compiled code and define the correctness of compiled code with respect to pattern-matching behavior. This allows 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 [64], 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 construct. This prototype implementation will be extended to support full multi-match constructs and to automatize the cooperation with Zenon.

The extension of this method to matching with theories, like associative matching or AC-matching is not a trivial task, since there are many new problems to solve, like proving that the matching-code enumerates all possible matching solutions, and formalizing these enumerations.

6.2.5. Rewriting, business rules and RETE network

Participants: Horatiu Cirstea, Claude Kirchner, Pierre-Etienne Moreau, Michael Moossen.

Most of business rule systems, and the Ilog Rule system in particular, use the RETE algorithm to discriminate the rules which have to be applied and fired. In the project, we have developed several compilation algorithms based on the syntactic structure of the rules (e.g. [70]).

We have proposed a new formalization of the RETE algorithm [45] as well as a first formalization of the relationship between these two approaches: the classical term-based indexing algorithm and the RETE based production systems [46]. We recently started to develop a formal basis for integrating rewriting and constraint solving. This should become the theoretical foundation of a more expressive rule based language where constraints could be used to improve the expressivity of the patterns and the actions used in business rules.

6.2.6. XML, XQuery, and rewriting

Participants: Horatiu Cirstea, Guillaume Darmont, Claude Kirchner, Pierre-Etienne Moreau, Nghiem Long Phan.

In order to improve the expressiveness of our tools, we have studied the integration of XML pattern matching facilities, as well a static typing support, into the *Tom* system. This has been done in the internship of Guillaume Darmont [62]..

During the internship of Nghiem Long Phan, a comparison of XQuery and *Tom* has been done. This has been followed by a work on systematic translation from XQuery to *Tom* programs.

6.2.7. Types for XML

Participants: Horatiu Cirstea, Claude Kirchner, Pierre-Etienne Moreau, Anderson Santana, Benjamin Wack.

We have contributed to the survey and analyse of the relevant existing works on typing of rules, in particular on typing of constraint logic programs and discuss applicability of these approaches to the reasoning and query languages under development in the REWERSE network of excellence such as XPathLog and Xcerpt [44].

6.2.8. Manipulating and executing algebraic specifications

Participants: Stéphane Hrvatin, Anamaria Martins Moreira, Christophe Ringeissen.

In an environment of continuous and rapid evolution, software design methodologies must incorporate techniques and tools that support changes in software artifacts. In the FERUS project, we have developed a tool targeted at software designers that integrates a collection of operations on algebraic specifications written in the CASL language. The scope of FERUS includes not only modification of existing specifications, but also creation or derivation of new specifications, as well as their proof and execution, which are realized through inter-operability with existing tools. As FERUS involves the manipulation of software specification and inter-operability with other tools, the question of choosing appropriate representation formats is important. In [17], we discuss the advantages and limitations of ATerms as a manipulation and exchange format in the setting of FERUS. We also present a new, graph-like format, which offers complementary features to a term-based format. Moreover, we present visualization utilities for these formats. This is a joint work with David Déharbe and Gleydson Lima, from University of Natal (Brazil). This work was partly supported, until 2003, by a joint INRIA-CNPq project.

In connection with the FERUS project, we studied how to write structured specifications using the structuring mechanisms of CASL and the *Tom* syntax for basic specifications. This study has led to the development of a prototype to execute CASL-like structured *Tom* specifications. This prototype makes use of the CASL Tool Set developed by the CoFI initiative to translate structured specifications into basic ones. Then, *Tom* is used to execute basic specifications [68].

6.3. Constraints

Keywords: *constraints*, *decision procedures*.

We are studying the design of symbolic constraint solvers and decision procedures. In particular, we focus on the combination problems for satisfiability procedures.

6.3.1. A rule language for interaction

Participants: Carlos Castro, Christophe Ringeissen.

In collaboration with Eric Monfroy from IRIN at University of Nantes, we propose in [27] a rule language to design interactive component languages and basic coordination languages. In our language, concurrent rules manage interactions of call-back functions that apply on a store of data. This store can freely be structured as an array, a list, a set of communication channels, etc. Our rule language can be seen as an abstract machine to specify and implement interactive component languages. We also propose such a component language devoted to solver cooperations and solver cooperation languages. We illustrate the use of this specific component language to implement some primitives of an existing solver cooperation language.

6.3.2. Combination of Nelson-Oppen and Shostak theories

Participants: Hélène Kirchner, Christophe Ringeissen, Duc Tran.

In collaboration with Silvio Ranise from the Cassis project, we consider the problem of building satisfiability procedures for unions of disjoint theories [36]. We briefly review the combination schemas proposed by Nelson-Oppen, Shostak, and others. Three inference systems are directly derived from the properties satisfied

by the theories being combined and known results from the literature are obtained in a uniform and abstract way. This rational reconstruction is the starting point for further investigations. We introduce the concept of extended canonizer and derive a modularity result for a new class of theories (larger than Shostak and smaller than Nelson-Oppen theories) which is closed under disjoint union. This is in contrast with the lack of modularity of Shostak theories. We also explain how to implement extended canonizers by using the basic building blocks used in Shostak schema or by means of rewriting techniques.

6.4. Rewriting and strategies

Keywords: *rules*, *strategies*.

This section considers how to extend the notion of rewriting controlled by strategies, In particular, we continue the study of probabilistic strategies.

6.4.1. Rewriting and probabilities

Participants: Olivier Bournez, Florent Garnier, Claude Kirchner.

In [56], we presented an abstract way to implement probabilistic strategies in rule based languages. No real implementation have been done at that time. This year, Florent Garnier worked in order to implement the proposed probabilistic strategies in ELAN 4.

Using the basic probabilistic strategy operators, we modeled in ELAN 4 the protocols CSMA-CA and CSMA-CD.

We also started to work on techniques for proving almost sure termination of a set of rewrite rules, in particular on techniques that ensure that the mean time before termination is always finite. These techniques are presented in the technical report [42], and have been submitted to a conference.

6.4.2. Rules and strategies for generation of chemical reactions

Participants: Olivier Bournez, Liliana Ibănescu, Hélène Kirchner.

Liliana Ibănescu defended her PhD thesis on "Rule-Based Programming and Strategies for Automated Generation of Detailed Kinetic Models for Gas Phase Combustion of Polycyclic Hydrocarbon Molecules" [12] on June 14th.

The primary objective of this thesis is to explore the approach of using rule-based systems and strategies, for a complex problem of chemical kinetic: the automated generation of reaction mechanisms. The chemical reactions are naturally expressed as conditional rewriting rules. The control of the chemical reactions chaining is easy to describe using a strategies language, such as the one of the *ELAN* system.

The thesis presents the basic concepts of the chemical kinetics, the chemical and computational problems related to the conception and validation of a reaction mechanism, and gives a general structure for the generator of reaction mechanisms called *GasEl*. Our research focuses on the primary mechanism generator. We give solutions for encoding the chemical species, the reactions and their chaining, and we present the prototype developed in *ELAN*. The representation of the chemical species uses the notion of molecular graphs, encoded by a term structure called *GasEl* terms. The chemical reactions are expressed by rewriting rules on molecular graphs, encoded by a set of conditional rewriting rules on *GasEl* terms. The strategies language of the *ELAN* system is used to express the reactions chaining in the primary mechanism generator. This approach is illustrated by coding ten generic reactions of the oxidizing pyrolysis. Qualitative chemical validations of the prototype show that our approach gives, for acyclic molecules, the same results as the existing mechanism generators, and for polycyclic molecules produces original results.

This thesis has been awarded by one of the INPL best PhD prices.

6.4.3. Rule based programming and verification

Participants: Horatiu Cirstea, Chandan Dubey, Claude Kirchner, Pierre-Etienne Moreau, Antone Reilles.

Strategic programming [33] has been developed both for filtering information in the internship of Chandran Dubey and for verification.

Indeed, in [30], we have presented an approach for the development of model-checkers in *Tom*, merging declarative and imperative features. We illustrated our method by specifying the Needham-Schroeder public-key protocol that aims at establishing a mutual authentication between an initiator and a responder that communicate via an insecure network. We described the behavior of the agents exchanging messages as well as the intruders and the security invariants the protocol should verify using the rewrite rules of *Tom*. The (depth-first or breadth-first) exploration of the search space was described using the imperative features of the language. We also proposed several optimizations and we compared our results to existing approaches.

Recently, in collaboration with Ozan Kahramanogullari we worked on the calculus of structures and proposed a flexible implementation in Tom. This allows to perform various kind of experimentations and should help to find a good proof search strategy.

6.5. Mechanized deduction

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

We are working on integrating first-order and higher-order deduction, calculi, decision procedures and constraint solving. We also develop proof search procedures for program properties.

On the one hand, this year, we have designed a general proof-theoretic setting where completion-like processes can be modeled and studied; we have described a superposition calculus where quantifiers are eliminated lazily. We also have studied further the links between proof search, representation and check through the deduction modulo and the ρ-calculus.

On the other hand, we have developed new techniques for property proofs of rewriting with strategies, and especially, we have studied termination of rewriting with strategies.

6.5.1. Size-based termination

Participant: Frédéric Blanqui.

In our previous works on the termination of rewrite-based function definitions in the Calculus of Constructions, a powerful system with polymorphic and dependent types, we could not capture definitions with recursive calls on non-structurally smaller arguments. Yet, since 1996, several authors (Hughes-Pareto-Sabry, Giménez, Abel, Barthe, etc.) devised size-based termination criteria for ML-like languages with fixpoint and case constructions allowing function definitions with non-structurally smaller recursive calls. All these works are based on the way inductive types are usually constructed: as fixpoints of monotone operators on the set of terms [14]. The number of steps necessary for reaching a term in such a fixpoint gives us a simple and natural measure, the "size", which is finer than the syntactic structure. By enriching the syntax of types with size annotations, we can express that a term has a size (strictly) smaller than another term. In [21], we extended this notion to rewriting and dependent types, and proved that it provides a powerful termination criterion.

6.5.2. Certification of termination proofs

Participants: Frédéric Blanqui, Sébastien Hinderer.

For his master thesis, Sébastien Hinderer studied the certification of termination proofs based on polynomial interpretations [48]. This work was part of the Averroes project. The idea was to certify, by building a Coq proof [61], the results of termination tools like CiME [60], AProVE, etc. To this end, a Coq library on polynomials with multiple variables has been developed, and the termination criterion used in those tools has been formalized in Coq (given a polynomial interpretation for function symbols, a rewrite system is terminating if, for every rule, the interpretation of the left hand-side is greater than the interpretation of the right hand-side). Then, tactics have been developed in Coq for automatically checking the conditions of this criterion. A small extension of the current Coq version has also been developed in order to declare rewrite rules and call the termination tool CiME for finding a polynomial interpretation, whose result is then checked inside Coq.

6.5.3. Higher-order conditional rewriting

Participants: Frédéric Blanqui, Claude Kirchner, Colin Riba.

With the beginning of Colin's PhD in October, we started to study the properties (confluence, termination, etc.) that are preserved when combining conditional rewriting and β-reduction. Our ultimate goal is to integrate in a nice and coherent way membership equational logic and type theory. This would provide a very powerful and elegant mechanism for defining inductive types and functions.

6.5.4. Abstract canonical presentations

Participants: Romain Demangeon, 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 centers around well-founded orderings of proofs. It allows for abstract definitions and very general characterizations of saturation processes and redundancy criteria [63]. The internship of Romain Demangeon consisted to instantiate the general framework to equational completion, ground or not.

6.5.5. 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 [47], we are working on understanding the relationship between induction based on specific models (like the natural numbers) and noetherien induction based on terms. For this we need to better understand when a reduction ordering could be used intead of a simplification ordering. We provide first sufficient conditions for this.

6.5.6. Inductive termination proofs

Participants: Olivier Fissore, Isabelle Gnaedig, Hélène Kirchner.

In our previous work, we have proposed a new method for specifically proving, by explicit induction, termination of rewriting under strategies. For a given term rewriting system, we establish on the ground term algebra that every term terminates i.e. has only finite derivations, by supposing that any smaller term terminates. For that, we develop 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 schematizes normalization of subterms the induction hypothesis is applied on, and narrowing, schematizing rewriting of the resulting form 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.

As it is lying on proof trees representing rewriting trees, our proof process is also well suited for the weak termination proof, we have focused on this year. Indeed, weak termination requires that at least one computation branch starting from each term is finite. So we have adapted our method to just develop branches of the proof trees, that represent one terminating branch of the rewriting tree of any ground term. We proceed by developing the trees in a breadth-first manner, and by cutting redundant branches, that represent in fact different ways of rewriting the same ground terms. Redundancy is detected in comparying the most general unifiers of a same narrowing step in the proof process [31].

This is the first method for specifically establishing weak termination of rewriting. It is of interest since rule-based programs often do not strongly terminate, but only weakly terminate. Moreover, in languages like ELAN, strategies exist to express that the result of a program evaluation on a data is precisely one of its possible finite evaluations.

The constructive aspect of our proof method, identifying and memorizing the terminating branches in the proof trees, allows us in addition to find a finite evaluation without fail for every term. The second interest

of this approach is then to increase the efficiency of evaluation: performing the proof process at compile-time of a program enables to adopt a depth-first strategy at run-time for evaluation, avoiding by this way costly breadth-first evaluations. This work has been realized in the context of the PRST-IL project Toundra.

6.5.7. A toolbox for verification and proofs of rule-based program properties

Participants: Isabelle Gnaedig, Laika Moussa.

Our work on toolboxes for verification and proof of rule-based program properties, has been continued. A new implementation of Kounalis completeness verification procedure has been made in *Tom*. Lying on pattern generation from left-hand sides of rules, and matching, it enables to verify whether, for a given rule-based program, the definition of the operations is complete.

A new prototype of toolbox, has been designed and implemented, to integrate CARIBOO and the previous completeness procedure, with a user-friendly specification editor. This work has been realized in the context of the ACI-SI project Modulogic.

6.6. Computability and complexity

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

We focus on computational models working over reals with a discrete and continuous time. Concerning discrete time models, we mainly focus on the model of computation introduced by Blum Shub and Smale over the reals, later on extended over arbitrary structures by Poizat. We have obtained several syntactic characterizations of complexity classes in this model. These characterizations subsume classical ones when restricted to booleans. Concerning continuous time models, we have provided a machine independent characterization of elementary computable functions over reals that extends the characterization obtained by Campagnolo. The previous known characterization was restricted to functions over the reals preserving integers.We also shown that it is possible to build a minimization schema in order to get a characterization of all (non-necessarily elementary) computable functions preserving integers.

6.6.1. Complexity in the Blum, Shub, Smale model of computation

Participants: Olivier Bournez, Paulin Jacobé de Naurois.

In order to model discrete time computation over real numbers, Blum, Shub and Smale have introduced in 1989 a new model of computation, referred as BSS machine or sometimes real Turing machine. In this model, the complexity of a computational problem is given by the number of elementary arithmetical operations and comparisons needed to solve this problem, independently of the underlying representation of real numbers. This makes it very different from the recursive analysis model, and occurs to be a very natural way to describe computational problems over real numbers. The BSS model of computation has been later on extended to the notion of computation over arbitrary logical structure. Since classical complexity theory occurs to be the restriction of this general model to boolean structures, it gives a new insight on previous questions on classical complexity theory and its links with logic.

Paulin de Naurois' PhD thesis, jointly supervised by Olivier Bournez, Jean-Yves Marion (Calligramme research team) and Felipe Cucker (Hong Kong City University), is focused on studying complexity in this model. It has been defended on 15th of December, 2004 [13].

In this thesis, we have provided new completeness results for geometrical problems when the arbitrary structure is specialized to the set of real numbers with addition and order. The range of these completeness results covers many of the most important complexity classes over this setting, which is one of the three major ones for complexity theory over arbitrary structures.

We also provided several machine independent characterizations of complexity classes over arbitrary structures. We extended some results by Grädel, Gurevich and Meer in descriptive complexity, characterizing deterministic and non deterministic polynomial time decision problems in terms of logics over meta-finite structures. We extended some results in implicit complexity by Bellantoni and Cook [52], characterizing functions

computable in sequential deterministic polynomial time, and by Leivant and Marion [71], characterizing functions computable in parallel deterministic polynomial time in terms of algebras of recursive functions. We also provided some characterizations of functions computable within the polynomial hierarchy, extending a result by Bellantoni [51] and in polynomial alternating time. These results were published in [22], [41], [15], [23].

6.6.2. Analog computations

Participants: Olivier Bournez, Emmanuel Hainry.

There are many ways to model analog computers. Unifying those models would therefore be a crucial matter as opposed to the discrete case, there is no property stating that those models are equivalent. It is possible to distinguish two groups in those 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 [73]. 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.

In [24][43] and [25], we presented an *analog* and *machine-independent* algebraic characterization of elementarily computable functions over the real numbers in the sense of recursive analysis: we prove that they correspond to the smallest class of functions that contains some basic functions, and closed by composition, linear integration, and a simple limit schema.

We generalized this result to all higher levels of the Grzegorczyk Hierarchy.

Concerning *recursive analysis*, our results provide machine-independent characterizations of natural classes of computable functions over the real numbers, allowing to define these classes without usual considerations on higher-order (type 2) Turing machines. Concerning *analog models*, our results provide a characterization of the power of a natural class of analog models over the real numbers.

In [26], we presented an *analog* and *machine-independent* algebraic characterization of (all, i.e. non-necessarily elementarily) computable functions over the real numbers that preserve integers: we prove that they correspond to the smallest class of real functions containing some basic functions and closed by composition, linear integration and a very natural unique minimization schema.

In some sense, the problem we solve there is the definition of a minimization operator, which is strong enough to get at least Turing machine power, but not too strong to get the technical problems of [74], nor non-robust super-Turing Zeno phenomena of [49][55][65][67][74].

7. Contracts and Grants with Industry

7.1. Averroes

Participants: Frédéric Blanqui, Olivier Bournez, Horatiu Cirstea, Claude Kirchner, Anamaria Martins Moreira, Huy Nguyen.

The main goal of the National Network on Software Technology (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”), 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 technolgies”), about adding rewriting to *Coq* [38][39][40][48], and about the mechanization of deduction.

7.2. GasEl

Participants: Olivier Bournez, Liliana Ibănescu, Hélène Kirchner.

The objectives of the *GasEl* project are to use rewriting and rule-based programming controlled by strategies for a specific problem in physical chemistry: the automated generation of combustion reaction kinetic mechanisms. The partners in this project are INPL, CNRS and **Peugeot Citroën Automobiles**. The implied laboratories are the LORIA and the Department of Chemistry and Physics of Reactions (Nancy).

For computer scientists, the scientific objectives are to use the rewriting systems and rule-based programming controlled by strategies for a very challenging problem, design a new software package, provide a prototype for the core system and to validate it.

For chemists, the scientific objectives are to extend their knowledge about detailed mechanisms for hydrocarbons molecules with cycles, use the new software package to elaborate and validate detailed oxidation and combustion mechanisms for cyclic hydrocarbon molecules.

For the industrial partner, the scientific objectives is to use the detailed mechanisms for hydrocarbons molecules with cycles in order to change the fuel composition and review the conception/design of engines.

The project was initiated with the PhD thesis of Liliana Ibănescu, defended in June. The current state of the prototype and of the project at this date is presented in Liliana’s PhD document.

7.3. Manifico

Participants: Horatiu Cirstea, Claude Kirchner, Pierre-Etienne Moreau, Michael Moossen.

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

CPER Planning Contract between the Government and the Region
PRST Pole of Scientific and Technological Research of the CPER
PRST-IL PRST devoted to Software Intelligence
QSL PRST-IL project on Quality and Safety of Software
GDR CNRS Research Group
GDR-ALP GDR on Algorithmics, Languages and Programming
FNS National Fund for Science
ACI FNS Concerted and Incentive Action
ACI-SI ACI on Computer and Software Security
ESPRIT Information Technologies Program of the European Union
Compulog ESPRIT network on Computational Logic
ERCIM European Research Consortium for Informatics and Mathematics
PAI Integrated Action Programme

8.2. Regional initiatives

On the level of the CPER 2000-2006, we are involved in the PRST on Software Intelligence 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, and with the VALDA project on the automated verification of software in cooperation with Cassis.

8.2.1. Toundra

Participants: Olivier Fissore, Isabelle Gnaedig, Claude Kirchner, 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.

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. Modulogic

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

Modulogic is an ACI-SI project started on September, 1st. 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.

8.4. 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

AirCube. We are involved in the INRIA associated team called "AirCube" (Rewrite Rule Research). The AirCube project is a strong cooperation between the Protheo group and the Generic Language Technology group[1] of Paul Klint and Mark van den Brand at CWI in Amsterdam. It is based on a close relationship of the scientific interests of both groups and on the common goal to make rewriting concepts and tools widely available for program logic, semantics, and transformation. The two teams share a strong core of common knowledge and know-how in the field of rewriting and their collaboration benefits from complementary point of views on concepts, implementation, applications, and potential transfers. In addition to the construction of a shared knowledge of people and scientific as well as technological information, the scientific outcomes of the AirCube project are presented in Section 6.2.

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.

Hong Kong. Paulin de Naurois carries out his PhD thesis in co-supervision with the Hong Kong *City University*. His supervisors are Felipe Cucker (Hong Kong City University), Olivier Bournez and Jean-Yves Marion (Calligramme project). The PhD subject relies on the complexity in the Blum Shub and Smale model, and its connection with implicit complexity.

Udine and **Venizia.** Clara Bertolissi carries out her PhD thesis in co-supervision with the Universities of Udine and Venizia. The supervisors are Furio Honsell at Udine and Paolo Baldan at Venizia (Italy) and Claude Kirchner, and Horatiu Cirstea at Nancy.

Lisbon. Olivier Bournez and Emmanuel Hainry cooperate with Manuel Campagnolo and Daniel Graça from Lisbon. An application to a PAI Pessoa grant has been submitted this year.

[1] http://www.cwi.nl/projects/MetaEnv

8.6. Visiting scientists

- Anamaria Martins Moreira (UFRN Natal, Brazil), carried out a one month visit.
- Gopalan Nadathur (Technology University of Minnesota, USA), carried out a 6 months visit finishing in March.
- Our french-chilean cooperation allows regular visits of Carlos Castro of the Technical University Federico Santa Maria, Valparaiso.
- Ozan Kahramanoğullari (Technische Universität Dresden, Germany), carried out a one month visit.

8.7. Invited lecturers

- Ozan Kahramanoğullari (TU Dresden), “Implementing deep inference”
- Stéphane Lengrand (PPS, Jussieu), “Barendregt’s Cube in sequent calculus”
- Jean-Yves Moyen (Calligramme, LORIA), “Termination by resources”
- Alexandre Miquel (PPS, Jussieu), “Semantics of the Calculus of Constructions”
- Sylvain Salvati (Calligramme, LORIA), “Higher-order matching in linear lambda-calculus”
- Gopalan Nadathur (Minnesota, USA), “The suspension notation of explicit substitutions”
- Julien Cohen (LAMI, Évry), “Typing programs with matching on topological collections”.
- Colin Riba (ENSIMAG, Grenoble), “Réécriture de graphes avec redirection de pointeurs”.

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

9. Dissemination

9.1. Leadership within scientific community

CADE International Conference on Automated Deduction

CSL Conference of the European Association for Computer Science Logic

CoSolv Workshop on Cooperative Solvers

IEHSC International Embedded and Hybrid Systems Conference

IFIP International Federation for Information Processing

IFIP-WG IFIP Working Group

IJCAI International Joint Conference on Artificial Intelligence

IJCAR International Joint Conference on Automated Reasoning

JFLA French-speaking workshop on applicative languages

JFPLC French-speaking Conference on Logic and Constraint Programming

LDTA Language Descriptions, Tools and Applications

LICS International Conference on Logics in Computer Science

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

WRLA International Workshop on Rewriting Logic and its Applications

WRS Workshop on Rewriting Strategies

SPECIF French society of professors and researchers in computer science

AFIT French chapter of EATCS

- Frédéric Blanqui:
  - Organization of the 1st Nancy-Saarbrücken Workshop on Logic, Proofs and Programs at Nancy, 17-18 June, 37 participants.
  - Organization of the 2nd Workshop on Coq and Rewriting, together with a GDR ALP meeting, at École Polytechnique, Palaiseau, 22-24 June, 52 participants.
- Olivier Bournez:
  - Leader of the french node of the "Computability in Europe" network.
  - Member of the scientific committee of IEHSC'05.
- Germain Faure
  - Organization of the first workshop on the rewriting calculus. Nancy, 11-12 March, 22 participants http://rho.loria.fr/workshop2004.html
- Isabelle Gnaedig:
  - Coordination board of the PRST-IL project on Quality and Safety of Software.
  - Coordinator of the Protheo part of the ACI-SI project Modulogic.
  - Manager of the PRST-IL project Toundra.
  - Program committee of JFPLC'04.
- Claude Kirchner:
  - Chair of the scientific committee for the national ACI-SI program [16].
  - Director of education through research at INRIA.
  - Chair of the LORIA building extension committee.
  - Editorial boards of *Journal of Automated Reasoning*, *Journal of Theory and Practice of Logic Programming*, *Journal of Information Science and Engineering*.
  - Program committees of IJCAR'04, WRLA'04 and of PPDP'04.
  - Chair of the IFIP-WG 1.6 on rewriting and applications.
  - Member of the advisory boards of QPQ, LICS and PPDP.
  - Co-organizer of the first French Taiwanese Conference in Information Technologies. April, 14-16.
  - President of the scientific committees of the Celar-SSI (Bruz) and of the LSV (Cachan).
  - Member of working groups of the CSD and the CISSI.

- Hélène Kirchner:
  - Director of LORIA and INRIA Lorraine.
  - Coordinator of the PRST on Software Intelligence.
  - Editorial boards of *Annals of Mathematics and Artificial Intelligence* and *Computing and Informatics*.
  - Editorial board of the QPQ forum on rewriting.
  - Member of the IFIP-WG 1.3 on foundations of systems specifications and the IFIP-WG 1.6 on term rewriting.
  - Orientation committee of National Network on Software Technology (RNTL).
  - Evaluation committee of the Institute for Computer Science at the University of Munich.
  - Program committees of RTA'04 and ASIAN'04.
  - Steering committees of PPDP and RTA.
  - Member of Scientific directorate of the Dagstuhl international conference center.
- Pierre-Etienne Moreau:
  - LORIA committee for computer and software equipments.
  - Program committees of WRLA'04 and JFLA'04.
  - Steering committee of LDTA.
  - Organization of an AirCube workshop (2-6 January) and two Modulogic workshops (14-17 September and 27-29 October)
- Christophe Ringeissen:
  - Correspondent for the european projects at LORIA.
  - Co-chair of RULE'04.
  - Member of the program committee of CoSolv'04.

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:
  - DEA course on algorithmic verification at UHP.
  - DEA course on computational complexity at UHP.
  - Master course on algorithmic and computational complexity at UHP.
  - Supervised practical works on algorithmics of parallel and distributed systems to ESIAL 2nd year students.
- Horatiu Cirstea:
  - Course on the rewriting calculus, with Claude Kirchner and Luigi Liquori, 16th European Summer School in Logic, Language and Information, Nancy.
- Isabelle Gnaedig:
  - Coordinator of the courses on program and software specification at ESIAL and UHP (for DESS diploma).
  - Courses on algebraic specifications, the LOTOS language and the semantic of concurrent processes at ESIAL and UHP.
  - Course and supervised practical works on formal methods for specifying and validating software to 2nd year students of the engineering school “École des Mines”.
- Claude Kirchner:
  - DEA course in Nancy on logic and automated theorem proving with Adam Cichon.
  - Course on Deduction Modulo at the ICCL Summer School 2004 on Proof Theory and Automated Theorem Proving, Dresden.
  - Course on the rewriting calculus, with Horatiu Cirstea and Luigi Liquori, 16th European Summer School in Logic, Language and Information, Nancy.
- Pierre-Etienne Moreau:
  - 6 hours lecture at ESIAL on fundamental data-structures.
- Hélène Kirchner:
  - 10 hours lecture at Sup’Com (Tunis) on computation and deduction.
- Christophe Ringeissen:
  - DEA course in Nancy on constraint solving and combinatorial optimization with An Le Thi Hoai.

9.3. Invited talks

- Olivier Bournez:
  - "On the computational power of some analog models", IST seminar, Lisbon, Portugal, March 19th.
  - "On matrix mortality", LLAIC1, Clermont-Ferrand I University, February 26th.
- Isabelle Gnaedig:
  - "Terminaison des programmes par règles", ACI Modulogic, Paris, March.
- Claude Kirchner:
  - "Faciliter la vérification de programmes existants par l'intégration d'îlots formels", 11e Rencontre INRIA-Industrie "L'ingénierie du logiciel", Rocquencourt, January.
  - "Protheo@loria Position Statement", Rewerse Kick-off meeting, Munich, March.
  - "Strategic Rewriting", WRS'04, Aachen, June.
  - "Recent developments of the rewriting calculus", SRI International, July.
  - "L'ACI Sécurité & Informatique", FING, Paris, September.
  - "Rule-based programming and proving: the ELAN experience outcomes", ASIAN'04, Chiang Mai, Thailand, December.
- Hélène Kirchner:
  - "Termination of rewriting under strategies" SRI International, July.
- Pierre-Etienne Moreau:
  - "Programming with rules", TAB ILOG, Paris.
  - "Compilation of AU-matching in Tom", Workshop Coq+Rewriting, LIX, Paris.

9.4. Visits

- Olivier Bournez
  - One week at the IST of Lisbon (Portugal) in March 2004.
- Claude Kirchner:
  - One week at SRI International (Menlo Park, USA).
- Hélène Kirchner:
  - One week at SRI International (Menlo Park, USA).

9.5. Thesis and admission committees

- Olivier Bournez:
  - AFIT PhD thesis award committee.
  - UHP recruitment committee (section 27, tenured since September).
  - INPL recruitment commitee (section 27, substitute) since September.
  - Metz University recruitment committee (section 27, substitute) since September.
  - Liliana Ibănescu, PhD Thesis, INPL.
  - Paulin de Naurois, PhD Thesis, City University Hong Kong.
- Frédéric Blanqui:
  - PhD thesis SPECIF award committee.
- Isabelle Gnaedig:
  - Admission committee ESIAL.
- Claude Kirchner:
  - Recruitment committees (section 27) of INPL and Metz University (until June). Recruitment committee for CR INRIA at Rocquencourt.
  - Didier Le Botlan, “MLF : Une extension de ML avec polymorphisme de second ordre et instanciation implicite. ”
  - Kristina Striegnitz “Generating Anaphoric Expressions: Contextual Reasoning in Sentence Planning.”
- Hélène Kirchner:
  - Recruitment committees (section 27) of UHP Nancy1, Nancy2 University, ULP Strasbourg (until August), INPL (since September).
  - Liliana Ibanescu, PhD Thesis, INPL.
- Pierre-Etienne Moreau:
  - Julien Cohen, “Intégration des collections topologiques et des transformations dans un langage fonctionnel”, PhD thesis, University of Evry.
  - Member of the UHP recruitment committee (section 27).

10. Bibliography

Major publications by the team in recent years

[1] F. BLANQUI. *Definitions by Rewriting in the Calculus of Constructions*, in "Mathematical Structures in Computer Science", to appear, Apr 2003.

[2] O. BOURNEZ, E. HAINRY. *An Analog Characterization of Elementarily Computable Functions Over the Real Numbers*, J. DIAZ, J. KARHUMÄKI, A. LEPISTO, D. T. SANNELLA (editors)., Lecture Notes in Computer Science, vol. 3142, Springer, Turku, Finland, 2004, p. 269-280.

[3] C. CASTRO. *Building Constraint Satisfaction Problem Solvers Using Rewrite Rules and Strategies*, in "Fundamenta Informaticae", vol. 34, nº 3, 1998, p. 263-293.

[4] 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, nº 3, May 2001, p. 427-498.

[5] G. DOWEK, T. HARDIN, C. KIRCHNER. *Theorem Proving Modulo*, in "Journal of Automated Reasoning", vol. 31, nº 1, Nov 2003, p. 33-72.

[6] O. FISSORE, I. GNAEDIG, H. KIRCHNER. *Outermost ground termination*, in "Proceedings of the Fourth International Workshop on Rewriting Logic and Its Applications, Pisa, Italy", Electronic Notes in Theoretical Computer Science, vol. 71, Elsevier Science Publishers B. V. (North-Holland), September 2002.

[7] C. KIRCHNER, H. KIRCHNER, M. VITTEK. *Designing Constraint Logic Programming Languages using Computational Systems*, in "Principles and Practice of Constraint Programming. The Newport Papers", V. SARASWAT, P. VAN HENTENRYCK (editors)., chap. 1, The MIT Press, 1995, p. 131-158.

[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", 2000.

[9] H. KIRCHNER, C. RINGEISSEN. *Combining Symbolic Constraint Solvers on Algebraic Domains*, in "J. Symbolic Computation", vol. 18, nº 2, August 1994, p. 113-155.

[10] C. KIRCHNER, C. RINGEISSEN. *Rule-Based Constraint Programming*, in "Fundamenta Informaticae", vol. 34, 1998, p. 225-262.

[11] 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.

Doctoral dissertations and Habilitation theses

[12] L. IBANESCU. *Programmation par règles et stratégies pour la génération automatique de mécanismes de combustion d'hydrocarbures polycycliques*, Thèse d'université, INPL, Jun 2004.

[13] P. JACOBÉ DE NAUROIS. *Completeness Results and Syntactic Characterizations of Complexity Classes over Arbitrary Structures*, Thèse d'université, INPL, Dec 2004.

Articles in referred journals and book chapters

[14] F. BLANQUI. *Inductive types in the Calculus of Algebraic Constructions*, in "Fundamenta Informaticae", Accepted for publication, Sep 2004.

[15] O. BOURNEZ, F. CUCKER, P. JACOBÉ DE NAUROIS, J.-Y. MARION. *Implicit Complexity Over an Arbitrary Structure : Sequential and Parallel Polynomial Time*, in "Journal of Logic and Computation", Nov 2004, http://www.loria.fr/publications/2004/A04-R-285/A04-R-285.ps.

[16] C. KIRCHNER. *L'Action Concerté Incitative Sécurité et Informatique*, in "Technique et Science Informatiques - TSI", vol. 23, n° 3, Apr 2004, p. 421-426.

[17] A. MARTINS MOREIRA, C. RINGEISSEN, D. DÉHARBE, G. LIMA. *Manipulating Algebraic Specifications with Term-based and Graph-based Representations*, in "The Journal of Logic and Algebraic Programming", Special Issue on Annotated Terms (ATerms), vol. 59, n° 1-2, Apr 2004, p. 63-87.

[18] P.-E. MOREAU, O. ZENDRA. *GC^2 : A Generational Conservative Garbage Collector for the ATerm Library*, in "The Journal of Logic and Algebraic Programming (JLAP)", vol. 59, n° 1-2, Apr 2004, p. 5-34.

[19] M. VAN DEN BRAND, P.-E. MOREAU, J. VINJU. *A generator of efficient strongly typed abstract syntax trees in Java*, in "IEE Proceedings - Software Engineering", Dec 2004.

Publications in Conferences and Workshops

[20] 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. FERNANDEZ (editor)., Electronic Notes in Theoretical Computer Science, Oct 2004.

[21] F. BLANQUI. *A type-based termination criterion for dependently-typed higher-order rewrite systems*, in "15th International Conference on Rewriting Techniques and Applications - RTA'04, Aachen, Germany", Jun 2004.

[22] O. BOURNEZ, F. CUCKER, P. JACOBÉ DE NAUROIS, J.-Y. MARION. *Tailoring Recursion to Characterize Non-Deterministic Complexity Classes Over Arbitrary Structures*, in "3rd IFIP International Conference on Theoretical Computer Science - TCS'2004, Toulouse, France", Kluwer Academic Press, Aug 2004, http://www.loria.fr/publications/2004/A04-R-287/A04-R-287.ps.

[23] O. BOURNEZ, F. CUCKER, P. JACOBÉ DE NAUROIS, J.-Y. MARION. *Tailoring Recursion to Characterize Non-Deterministic Complexity Classes Over Arbitrary Structures*, in "2nd APPSEM II Workshop - APPSEM'2004, Tallinn, Estonia", Apr 2004, http://www.loria.fr/publications/2004/A04-R-419/A04-R-419.ps.

[24] O. BOURNEZ, E. HAINRY. *An analog Characterization of Elementarily Computable Functions Over the Real Numbers*, in "31st International Colloqiuim 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.

[25] O. BOURNEZ, E. HAINRY. *An analog Characterization of Elementarily Computable Functions Over the Real Numbers*, in "2nd APPSEM II Workshop - APPSEM'2004, Tallinn, Estonia", Apr 2004, http://www.loria.fr/publications/2004/A04-R-289/A04-R-289.ps.

[26] 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.

[27] C. CASTRO, E. MONFROY, C. RINGEISSEN. *A Rule Language for Interaction*, in "Joint ERCIM/CoLogNet International Workshop on Constraint Solving and Constraint Logic Programming - CSCLP'03, Budapest, Hongrie", K. R. APT, F. FAGES, F. ROSSI, P. SZEREDI, J. VANCZA (editors)., Lecture Notes in Artificial Intelligence, published in post-workshop proceedings entitled "Recent Advances in Constraints", vol. 3010, Springer-Verlag, May 2004, p. 154-170.

[28] H. CIRSTEA, G. FAURE, C. KIRCHNER. *A rho-calculus of explicit constraint application*, in "5th International Workshop on Rewriting Logic and its Applications - WRLA'2004, Barcelona, Spain", N. MARTI-OLIET, M. CLAVEL, A. VERDEJO (editors)., Electronic Notes in Theoretical Computer Science, Elsevier, Narciso Marti-Oliet, Mar 2004.

[29] H. CIRSTEA, L. LIQUORI, B. WACK. *Rewriting Calculus with Fixpoints : Untyped and First-order Systems*, in "Annual Workshop on the Types Working Group - TYPES'2003, Torino, Italie", S. BERARDI, M. COPPO, F. DAMIANI (editors)., Lecture Notes in Computer Science, vol. 3085, Springer, Mar 2004, p. 147-161.

[30] H. CIRSTEA, P.-E. MOREAU, A. REILLES. *Rule based programming in Java for protocol verification*, in "5th International Workshop on Rewriting Logic and its Applications - WRLA'2004, Barcelona, Spain", N. MARTI-OLIET, M. CLAVEL, A. VERDEJO (editors)., Electronic Notes in Theoretical Computer Science, Elsevier, Narciso Marti-Oliet, Mar 2004.

[31] O. FISSORE, I. GNAEDIG, H. KIRCHNER. *A proof of weak termination providing the right way to terminate*, in "First International Colloquium on Theoretical Aspects of Computing, Guiyang, Chine", Lecture notes in Computer Science, Springer Verlag, Sep 2004.

[32] J. GUYON, P.-E. MOREAU, A. REILLES. *An Integrated Development Environment for Pattern Matching Programming*, in "2nd eclipse Technology eXchange workshop - eTX'2004, Barcelona, Spain", Electronic Notes in Theoretical Computer Science, Brian Barry and Oege de Moor, Apr 2004.

[33] C. KIRCHNER. *Strategic Rewriting*, in "4th International Workshop on Reduction Strategies in Rewriting and Programming - WRS'2004, Aachen, Germany", Jun 2004, http://www.loria.fr/publications/2004/A04-R-364/A04-R-364.ps.

[34] C. KIRCHNER, H. KIRCHNER. *Rule-based programming and proving : the ELAN experience outcomes*, in "Ninth Asian Computing Science Conference - ASIAN'04, Chiang Mai, Thailand", Dec 2004, http://www.loria.fr/publications/2004/A04-R-363/A04-R-363.ps.

[35] L. LIQUORI, B. WACK. *The Polymorphic Rewriting Calculus : Type checking vs. Type inference*, in "5th International Workshop on Rewriting Logic and its Applications - WRLA 2004, Barcelona, Spain", N.

MARTI-OLIET, M. CLAVEL, A. VERDEJO (editors)., Electronic Notes in Theoretical Computer Science, Elsevier, Narciso Marti-Oliet, Mar 2004, http://www.loria.fr/publications/2004/A04-R-386/A04-R-386.ps.

[36] S. RANISE, C. RINGEISSEN, D.-K. TRAN. *Nelson-Oppen, Shostak and the Extended Canonizer : A Family Picture with a Newborn*, in "First International Colloquium on Theoretical Aspects of Computing - ICTAC 2004, Guiyang, Chine", Lecture Notes in Computer Science, To appear in post-event proceedings, Springer-Verlag, Keijiro Araki, Zhiming Liu, Sep 2004.

[37] B. WACK. *The Simply-typed Pure Pattern Type System Ensures Strong Normalization*, in "3rd IFIP International Conference on Theoretical Computer Science - TCS'2004, Toulouse, France", J.-J. LÉVY (editor)., Kluwer Academic Publishers, Jean-Jacques Lévy, Aug 2004, p. 633-646.

Internal Reports

[38] F. BLANQUI. *Définition de la classe de réécriture à intégrer*, Rapport Intermédiaire, RNTL Averroes project report, Feb 2004, http://www.loria.fr/publications/2004/A04-R-487/A04-R-487.ps.

[39] F. BLANQUI. *Proposition d'architecture du moteur de test de conversion*, Rapport Intermédiaire, RNTL Averroes project report, Mar 2004, http://www.loria.fr/publications/2004/A04-R-488/A04-R-488.ps.

[40] F. BLANQUI. *Prototype d'extension du système Coq*, Rapport Intermédiaire, RNTL Averroes project report, Mar 2004, http://www.loria.fr/publications/2004/A04-R-505/A04-R-505.ps.

[41] O. BOURNEZ, F. CUCKER, P. JACOBÉ DE NAUROIS, J.-Y. MARION. *Implicit Complexity over an Arbitrary Structure : Quantifier Alternations*, Rapport de recherche, Loria, Oct 2004, http://www.loria.fr/publications/2004/A04-R-300/A04-R-300.ps.

[42] O. BOURNEZ, F. GARNIER. *Proving Positive Almost-Sure Termination*, Rapport de recherche, Loria, Nov 2004, http://www.loria.fr/publications/2004/A04-R-409/A04-R-409.ps.

[43] O. BOURNEZ, E. HAINRY. *Elementarily Computable Functions Over the Real Numbers and $\mathbb{R}$-Sub-Recursive Functions*, Rapport de recherche, Loria, Sep 2004, http://www.loria.fr/publications/2004/A04-R-301/A04-R-301.ps.

[44] H. CIRSTEA, E. COQUERY, W. DRABENT, F. FAGES, C. KIRCHNER, J. MALUSZYNSKI, B. WACK. *Types for Web Rule Languages : a preliminary study*, Rapport Intermédiaire, Sep 2004, http://www.loria.fr/publications/2004/A04-R-560/A04-R-560.ps.

[45] H. CIRSTEA, C. KIRCHNER, M. MOOSSEN, P.-E. MOREAU. *Production Systems and Rete Algorithm Formalisation*, Rapport Intermédiaire, Oct 2004, http://www.loria.fr/publications/2004/A04-R-546/A04-R-546.ps.

[46] H. CIRSTEA, C. KIRCHNER, M. MOOSSEN, P.-E. MOREAU. *Production Systems and Rewrite Systems*, Rapport Intermédiaire, Nov 2004, http://www.loria.fr/publications/2004/A04-R-563/A04-R-563.ps.

[47] E. DEPLAGNE, C. KIRCHNER. *Induction as Deduction Modulo*, Rapport de recherche, Nov 2004, http://www.loria.fr/publications/2004/A04-R-468/A04-R-468.ps.

[48] S. HINDERER. *Certification des preuves de terminaison par interprétations polynomiales*, Stage de DEA, Loria, Jun 2004, http://www.loria.fr/publications/2004/A04-R-489/A04-R-489.ps.

Bibliography in notes

[49] E. ASARIN, O. MALER. *Achilles and the Tortoise Climbing Up the Arithmetical Hierarchy*, in "Journal of Computer and System Sciences", vol. 57, n° 3, December 1998, p. 389–398.

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

[51] S. BELLANTONI. *Predicative recursion and the polytime hierarchy*, in "Feasible Mathematics II", P. CLOTE, J. REMMEL (editors)., Perspectives in Computer Science, Birkhäuser, 1994.

[52] S. BELLANTONI, S. COOK. *A new recursion-theoretic characterization of the poly-time functions*, in "Computational Complexity", vol. 2, 1992, p. 97-110.

[53] P. BOROVANSKY, C. KIRCHNER, H. KIRCHNER, P.-E. MOREAU. *ELAN from a rewriting logic point of view*, in "Theoretical Computer Science", n° 285, July 2002, p. 155-185.

[54] P. BOROVANSKÝ, C. KIRCHNER, H. KIRCHNER, C. RINGEISSEN. *Rewriting with strategies in ELAN: a functional semantics*, in "International Journal of Foundations of Computer Science", vol. 12, n° 1, February 2001, p. 69-98.

[55] O. BOURNEZ. *Complexité Algorithmique des Systèmes Dynamiques Continus et Hybrides*, Ph. D. Thesis, Ecole Normale Supérieure de Lyon, Janvier 1999.

[56] O. BOURNEZ, C. KIRCHNER. *Probabilistic rewrite strategies. Applications to ELAN*, in "13th International Conference on Rewriting Techniques and Applications - RTA'2002, Copenhagen, Denmark", S. TISON (editor)., Lecture Notes in Computer Science, vol. 2378, Springer, July 2002, p. 252-266.

[57] 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° 3, May 2001, p. 427–498.

[58] H. CIRSTEA, C. KIRCHNER, L. LIQUORI, B. WACK. *The rho cube : some results, some problems*, in "First International Workshop on Higher-Order Rewriting - HOR'02, Copenhague, Danemark", Held in conjunction with FLOC'02, D. Kesner, T. Nipkow & F. van Raamsdonk, July 2002.

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

[60] E. CONTEJEAN, C. MARCHÉ, B. MONATE, X. URBAIN. *CiME version 2*, 2000, http://cime.lri.fr/.

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

[62] G. DARMONT. *Transformation de documents XML dans un environnement typé*, 2004, Rapport de DEA, Université Henri Poincaré - Nancy 1.

[63] 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.

[64] D. DOLIGEZ. *Zenon: an automatic theorem prover for first-order logic*, Available as part of the Focal system at http://focal.inria.fr/download, 2004.

[65] G. ETESI, I. NÉMETI. *Non-Turing computations via Malament-Hogarth space-times*, in "International Journal Theoretical Physics", vol. 41, 2002, p. 341–370.

[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, October 2002, p. 62–73.

[67] M. L. HOGARTH. *Does General Relativity Allow an Observer to View an Eternity in a Finite Time?*, in "Foundations of Physics Letters", vol. 5, 1992, p. 173–181.

[68] S. HRVATIN. *Structuration pour les spécifications à base de règles : Etude et mise en œuvre pour TOM*, 2004, Rapport de DEA, Université Henri Poincaré - Nancy 1.

[69] 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.

[70] 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° 3, Mar 2001, p. 207-251.

[71] D. LEIVANT, J.-Y. MARION. *Ramified recurrence and computational complexity II: substitution and poly-space*, in "Computer Science Logic, 8th Workshop, CSL '94, Kazimierz,Poland", L. PACHOLSKI, J. TIURYN (editors)., Lecture Notes in Computer Science, vol. 933, Springer, 1995, p. 486-500.

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

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

[74] C. MOORE. *Recursion theory on the reals and continuous-time computation*, in "Theoretical Computer Science", vol. 162, n° 1, 5 August 1996, p. 23–44.

[75] P.-E. MOREAU, C. RINGEISSEN, M. VITTEK. *A Pattern Matching Compiler for Multiple Target Languages*, in "International Conference on Compiler Construction - CC'2003, Varsovie, Pologne", G. HEDIN (editor)., Lecture notes in Computer Science, vol. 2622, Apr 2003, p. 61-76.

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