INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Project-Team abstraction

Abstract Interpretation and Static Analysis

Paris - Rocquencourt

Theme : Programs, Verification and Proofs

Table of contents

1. Team

Research Scientists

Bruno Blanchet [Junior Researcher, CNRS — Feb. 2010[1], HdR]
Radhia Cousot [Senior Researcher, CNRS, HdR]
Jérôme Feret [Junior Researcher, INRIA Paris–Rocquencourt]
Antoine Miné [Junior Researcher, CNRS]
Xavier Rival [Junior Researcher, INRIA Paris–Rocquencourt]

Faculty Members

Patrick Cousot [Team leader, Professor, ENS, HdR]
Laurent Mauborgne [Associate Professor, ENS, HdR]

Technical Staff

Élodie-Jane Sims [Research engineer, INRIA, — Sept. 2010]

PhD Students

David Cadé [— Feb. 2010]
Liqian Chen [Mar. 2009 — Aug. 2010]
Vincent Laviron [Sep. 2009 —]
Jérémy Leconte [Sep. 2009 — Sep. 2010]
Matteo Zanioli [Sep. 2009 —]
Ferdinanda Camporesi [May. 2010 —]

Post-Doctoral Fellow

Julien Bertrane

Visiting Scientists

Roberto Giacobazzi [Università di Verona, Nov. 2010]
Andreas Podelski [University of Freiburg, Oct. 2010]
Miriam Paiola [Università di Padova, Feb. 2010]
Ben Smyth [University of Birmingham, Feb. 2010]

Administrative Assistants

Joëlle Isnard [Administrative Head DI, ENS]
Elisabeth Baque [INRIA]
Nathalie Abiola [INRIA, — Feb. 2010]

Others

Mehdi Bouaziz [M2 internship, ENS, March–July 2010]
Suzanne Renard [Research internship, École des Mines de Paris, Sep. 2010–Feb. 2011]

2. Overall Objectives

2.1. Overall Objectives

Software has known a spectacular development this last decade both in its scope of applicability and its size. Nevertheless, software design, development and engineering methods remain mostly manual, hence error-prone. It follows that complex software-based systems are unsafe and insecure, which is not acceptable in safety-critical or mission-critical applications. Intellectual and computer-based tools must therefore be developed to cope with the safety and security problems.

[1] Bruno Blanchet, David Cadé, and Miriam Paiola joined the Cascade project-team in March 2010, thus their yearly contributions are described in the report of the Cascade project-team.

The notions of *abstraction* and *approximation*, as formalized by the *abstract interpretation theory*, are fundamental to design, model, develop, analyze, and verify highly complex systems, from computer-based to biological ones. They also underlie the design of safety and security verification *tools*.

2.2. Highlights

Radhia Cousot co-chaired the 17th International Static Analysis symposium with Matthieu Martel (University of Perpignan), from the 14th to the 16th of September and edited the proceedings [37]. Three satellite workshops have been organised. Antoine Miné co-chaired the Second International Workshop on Numerical and Symbolic Abstract Domains, with Enric Rodríguez Carbonell (Technical University of Catalonia, Spain); Jérôme Feret co-chaired the First International Workshop on Static Analysis and Systems Biology, with Andre Levchenko (Johns Hopkins University, USA); and Xavier Rival co-chaired the First International Workshop on Tools for Automatic Program AnalysiS, with David Delmas (Airbus France).

3. Scientific Foundations

3.1. Abstract Interpretation Theory

The abstract interpretation theory [6], [48], [53] is the main scientific foundation of the work of the ABSTRACTION project-team. Its main current application is on the safety and security of complex hardware and software computer systems.

Abstract interpretation is a theory of sound approximation of mathematical structures, in particular those involved in the behavior of computer systems. It allows the systematic derivation of sound methods and algorithms for approximating undecidable or highly complex problems in various areas of computer science (semantics, verification and proof, model-checking, static analysis, program transformation and optimization, typing, software steganography, etc...).

3.2. Formal Verification by Abstract Interpretation

The *formal verification* of a program (and more generally a computer system) consists in proving that its *semantics* (describing "what the program executions actually do") satisfies its *specification* (describing "what the program executions are supposed to do").

Abstract interpretation formalizes the idea that this formal proof can be done at some level of abstraction where irrelevant details about the semantics and the specification are ignored. This amounts to proving that an *abstract semantics* satisfies an *abstract specification*. An example of abstract semantics is Hoare logic while examples of abstract specifications are invariance, partial, or total correctness. These examples abstract away from concrete properties such as execution times.

Abstractions should preferably be *sound* (no conclusion derived from the abstract semantics is wrong with respect to the program concrete semantics and specification). Otherwise stated, a proof that the abstract semantics satisfies the abstract specification should imply that the concrete semantics also satisfies the concrete specification. Hoare logic is a sound verification method, debugging is not (since some executions are left out), bounded model checking is not either (since parts of some executions are left out). Unsound abstractions lead to *false negatives* (the program may be claimed to be correct/non erroneous with respect to the specification whereas it is in fact incorrect). Abstract interpretation can be used to design sound semantics and formal verification methods (thus eliminating all false negatives).

Abstractions should also preferably be *complete* (no aspect of the semantics relevant to the specification is left out). So if the concrete semantics satisfies the concrete specification this should be provable in the abstract. However program proofs (for non-trivial program properties such as safety, liveness, or security) are undecidable. Nevertheless, we can design tools that address undecidable problems by allowing the tool not to terminate, to be driven by human intervention, to be unsound (e.g. debugging tools omit possible executions), or to be incomplete (e.g. static analysis tools may produce false alarms). Incomplete abstractions lead to *false positives* or *false alarms* (the specification is claimed to be potentially violated by some program executions while it is not). Semantics and formal verification methods designed by abstract interpretation may be complete (e.g. [51], [52], [56]) or incomplete (e.g. [2]).

Sound, automatic, terminating and precise tools are difficult to design. Complete automatic tools to solve non-trivial verification problems cannot exist, by undecidability. However static analysis tools producing very few or no false alarms have been designed and used in industrial contexts for specific families of properties and programs [54]. In all cases, abstract interpretation provides a systematic construction method based on the effective approximation of the concrete semantics, which can be (partly) automated and/or formally verified.

Abstract interpretation aims at:

- providing a basic coherent and conceptual theory for understanding in a unified framework the thousands of ideas, concepts, reasonings, methods, and tools on formal program analysis and verification [6], [53];
- guiding the correct formal design of *abstract semantics* [52], [56] and automatic tools for *program analysis* (computing an abstract semantics) and *program verification* (proving that an abstract semantics satisfies an abstract specification) [49].

Abstract interpretation theory studies semantics (formal models of computer systems), abstractions, their soundness, and completeness.

In practice, abstract interpretation is used to design analysis, compilation, optimization, and verification tools which must automatically and statically determine properties about the runtime behavior of programs. For example the ASTRÉE static analyzer (Section 5.1), which was developed by the team over the last decade, aims at proving the absence of runtime errors in programs written in the C programming language. It is used in the aerospace industry to verify very large, synchronous, time-triggered, real-time, safety-critical, embedded software.

3.3. Advanced Introductions to Abstract Interpretation

A recent, short, informal, and intuitive introduction to the theory of abstract interpretation can be found in [32], see also "Abstract Interpretation in a Nutshell" [2] on the web. A more comprehensive introduction is available online [3]. The paper entitled "Basic concepts of abstract interpretation" [50] and an elementary "course on abstract interpretation" [4] can also be found on the web.

4. Application Domains

4.1. Certification of Safety Critical Software

Safety critical software may incur great damage in case of failure, such as human casualties or huge financial losses. These include many kinds of embedded software, such as fly-by-wire programs in aircrafts and other avionic applications, control systems for nuclear power plants, or navigation systems of satellite launchers. For instance, the failure of the first launch of Ariane 5 (flight Ariane 501) was due to overflows in arithmetic computations. This failure caused the loss of several satellites, worth up to $ 500 millions.

[2] www.di.ens.fr/~cousot/AI/IntroAbsInt.html
[3] www.di.ens.fr/~cousot/AI/
[4] web.mit.edu/afs/athena.mit.edu/course/16/16.399/www/

This development of safe and secure critical software requires formal methods so as to ensure that they do not go wrong, and will behave as specified. In particular, testing or bug finding methods do not provide any guarantee that no failure will occur, even of a given type such as runtime errors; therefore, their scope is limited for certification purposes. For instance, testing can usually not be performed for *all* possible inputs due to feasibility and cost reasons, so that it does not prove anything about a large number of possible executions.

By contrast, program analysis methods such as abstract-interpretation-based static analysis are not subject to unsoundness, since they can *formally prove* the absence of bugs. Yet, these techniques are generally incomplete since the absence of runtime errors is undecidable. Therefore, in practice, they are prone to false alarms (*i.e.*, they may fail to prove the absence of runtime errors for a program which is safe). The objective of certification is to ultimately eliminate all false alarms.

It should be noted that, due to the size of the critical codes (typically from 100 to 1000 kLOCs), only scalable methods can succeed (in particular, software model checking techniques are subject to state explosion issues). As a consequence, this domain requires efficient static analyses, where costly abstractions should be used only parsimoniously.

Furthermore, many families of critical software have similar features, such as the reliance on floating-point intensive computations for the implementation of control laws, including linear and non-linear control with feedback, interpolations, and other DSP algorithms. Since we stated that a proof of absence of runtime errors is required, very precise analyses are required, which should be able to yield no false alarm on wide families of critical applications. To achieve that goal, significant advantages can be found in the design of domain specific analyzers, such as ASTRÉE [47], [55], which has been initially designed specifically for synchronous embedded software.

Last, some specific critical software qualification procedures may require additional properties being proved. As an example, the DO-178 regulations (which apply to avionics software) require a tight, documented, and certified relation to be established between each development stage. In particular, compilation of high level programs into executable binaries should also be certified correct.

The ABSTRACTION project-team has been working on both proof of absence of runtime errors and certified compilation over the decade, using abstract interpretation techniques. Successful results have been achieved on industrial applications using the ASTRÉE analyzer. Following this success, ASTRÉE has been licensed to AbsInt Angewandte Informatik GmbH to be industrialized, and the ABSTRACTION project-team has strong plans to continue research on this topic.

4.2. Abstraction of Biological Cell Signaling Networks

Protein-protein interactions consist in complexations and post translational modifications such as phosphorilation. These interactions enable biological organisms to receive, propagate, and integrate signals that are expressed as proteins concentrations in order to make decisions (on the choice between cell division and cell death for instance). Models of such interaction networks suffer from a combinatorial blow up in the number of species (number of non-isomorphic ways in which some proteins can be connected to each others). This large number of species makes the design and the analysis of these models a highly difficult task. Moreover the properties of interest are usually quantitative observations on stochastic or differential trajectories, which are difficult to compute or abstract.

Contextual graph-rewriting systems allow a concise description of these networks, which leads to a scalable method for modeling them. Then abstract interpretation allows the abstraction of these systems properties. First qualitative abstractions (such as over approximation of complexes that can be built) provide both debugging information in the design phases (of models) and static information that are are necessary in order to make other computations (such as stochastic simulations) scale up. Then qualitative invariants also drive efficient quantitative abstractions (such as the reduction of ordinary differential semantics).

The work of the ABSTRACTION project-team on biological cell signaling networks ranges from qualitative abstraction to quantitative abstraction.

5. Software

5.1. The Astrée Static Analyzer

Participants: Patrick Cousot [project leader, correspondent], Radhia Cousot, Jérôme Feret, Laurent Mauborgne, Antoine Miné, Xavier Rival.

The ASTRÉE static analyzer [47], [55] www.astree.ens.fr aims at proving the absence of runtime errors in programs written in the C programming language.

ASTRÉE analyzes structured C programs, with complex memory usages, but without dynamic memory allocation and recursion. This encompasses many embedded programs as found in earth transportation, nuclear energy, medical instrumentation and aerospace applications, in particular synchronous control/command. The whole analysis process is entirely automatic.

ASTRÉE discovers all runtime errors including:

- undefined behaviors in the terms of the ANSI C99 norm of the C language (such as division by 0 or out of bounds array indexing);
- any violation of the implementation-specific behavior as defined in the relevant Application Binary Interface (such as the size of integers and arithmetic overflows);
- any potentially harmful or incorrect use of C violating optional user-defined programming guidelines (such as no modular arithmetic for integers, even though this might be the hardware choice);
- user-defined assertions.

The analyzer performs an abstract interpretation of the programs being analyzed, using a parametric domain (ASTRÉE is able to choose the right instantiation of the domain for wide families of software). This analysis produces abstract invariants, which over-approximate the reachable states of the program, so that it is possible to derive an *over*-approximation of the dangerous states (defined as states where any runtime error mentioned above may occur) that the program may reach, and produces alarms for each such possible runtime error. Thus the analysis is sound (it correctly discovers *all* runtime errors), yet incomplete, that is it may report false alarms (*i.e.*, alarms that correspond to no real program execution). However, the design of the analyzer ensures a high level of precision on domain-specific families of software, which means that the analyzer produces few or no false alarms on such programs.

In order to achieve this high level of precision, ASTRÉE uses a large number of expressive abstract domains, which allow expressing and inferring complex properties about the programs being analyzed, such as numerical properties (digital filters, floating-point computations), boolean control properties, and properties based on the history of program executions.

ASTRÉE has achieved the following two unprecedented results:

- **A340–300.** In Nov. 2003, ASTRÉE was able to prove completely automatically the absence of any RTE in the primary flight control software of the Airbus A340 fly-by-wire system, a program of 132,000 lines of C analyzed in 1h20 on a 2.8 GHz 32-bit PC using 300 MB of memory (and 50mn on a 64-bit AMD Athlon 64 using 580 MB of memory).
- **A380.** From Jan. 2004 on, ASTRÉE was extended to analyze the electric flight control codes then in development and test for the A380 series. The operational application by Airbus France at the end of 2004 was just in time before the A380 maiden flight on Wednesday, 27 April, 2005.

These research and development successes have led to consider the inclusion of ASTRÉE in the production of the critical software for the A350. ASTRÉE is currently industrialized by AbsInt Angewandte Informatik GmbH and is commercially available.

5.2. Thésée

Participant: Antoine Miné [correspondent].

The THÉSÉE prototype is a fork of the ASTRÉE static analyzer (see 5.1) that adds support for analyzing parallel embedded C software.

THÉSÉE analyzes C programs composed of a fixed set of threads that communicate through a shared memory and synchronization primitives (mutexes, FIFOs, blackboards, etc.), but without recursion nor dynamic creation of memory, threads nor synchronization objects. THÉSÉE assumes a real-time scheduler, where thread scheduling strictly obeys the fixed priority of threads. Our model follows the ARINC 653 OS specification used in embedded industrial aeronautic software. Additionally, THÉSÉE employs a weakly-consistent memory semantics to model memory accesses not protected by a mutex, in order to take into account soundly hardware and compiler-level program transformations (such as optimizations). THÉSÉE checks for the same run-time errors as ASTRÉE.

Compared to ASTRÉE, THÉSÉE features: a new iterator to compute thread interactions, a refined memory abstraction that takes into account the effect of interfering threads, and a new scheduler partitioning domain. This last domain allows discovering and exploiting mutual exclusion properties (enforced either explicitly through synchronization primitives, or implicitly by thread priorities) to achieve a precise analysis.

THÉSÉE is currently being applied to analyze a large industrial avionic software: 1.6 MLines of C and 15 threads, completed with a 2500-line model of the ARINC 653 OS developed for the analysis. The analysis currently takes 14h on a 2.66 GHz 64-bit intel server using one core and generates around 7600 alarms, which proves the scalability of the approach. Ongoing work aims at reducing the number of alarms. THÉSÉE is introduced in [17] (§ VI) and [28]. The THÉSÉE prototype is closed source; it has been delivered to Airbus France.

5.3. The Apron Numerical Abstract Domain Library

Participants: Antoine Miné [correspondent], Bertrand Jeannet [team PopArt, INRIA-RA].

The APRON library is dedicated to the static analysis of the numerical variables of a program by abstract interpretation. Its goal is threefold: provide ready-to-use numerical abstractions under a common API for analysis implementers, encourage the research in numerical abstract domains by providing a platform for integration and comparison of domains, and provide a teaching and demonstration tool to disseminate knowledge on abstract interpretation.

The APRON library is not tied to a particular numerical abstraction but instead provides several domains with various precision versus cost trade-offs (including intervals, octagons, linear equalities and polyhedra). A specific C API was designed for domain developers to minimize the effort when incorporating a new abstract domain: only few domain-specific functions need to be implemented while the library provides various generic services and fallback methods (such as scalar and interval operations for most numerical data-types, parametric reduced products, and generic transfer functions for non-linear assignments). For the analysis designer, the APRON library exposes a higher-level API with C, C++, OCaml, and Java bindings. This API is domain-neutral and supports a rich set of semantic operations, including parallel assignments (useful to analyze automata), substitutions (useful for backward analysis), non-linear numerical expressions, and IEEE floating-point arithmetic.

The APRON library is freely available on the web at http://apron.cri.ensmp.fr/library under the LGPL license and is hosted at INRIAGForge. Packages exist for the Debian and Fedora Linux distributions. In order to help disseminate the knowledge on abstract interpretation, a simple inter-procedural static analyzer for a toy language is included. An on-line version is deployed at http://pop-art.inrialpes.fr/interproc/interprocweb.cgi.

The APRON library is developed since 2006 and currently consists of 130 000 lines of C, C++, OCaml, and Java. This year has seen the addition of a new binding, for the Java language.

Current external library users include the Proval/Démon team (LRI Orsay, France), the Analysis of Computer Systems Group (New-York University, USA), the Sierum software analysis platform (Kansas State University, USA), NEC Labs (Princeton, USA), EADS CCR (Paris, France), IRIT (Toulouse, France), ONERA (Toulouse, France), CEA LIST (Saclay, France), VERIMAG (Grenoble, France), ENSMP CRI (Fontainebleau, France), the IBM T.J. Watson Research Center (USA), the University of Edinburgh (UK).

5.4. The Arrayal array content analysis functor

Participants: Patrick Cousot [correspondent], Radhia Cousot.

ARRAYAL is a prototype abstract domain library functor for array content static analysis by parametric segmentation. It was included by Francesco Logozzo in MSR CLOUSOT, a language agnostic abstract interpretation-based static contract analyzer and checker for `.NET`, see Sect. 6.4.1.

5.5. Translation Validation

Participant: Xavier Rival [correspondent].

The main goal of this software project is to make it possible to certify automatically the compilation of large safety critical software, by proving that the compiled code is correct with respect to the source code: When the proof succeeds, this guarantees that no compiler bug did cause incorrect code be generated. Furthermore, this approach should allow to meet some domain specific software qualification criteria (such as those in DO-178 regulations for avionics software), since it allows proving that successive development levels are correct with respect to each other *i.e.*, that they implement the same specification. Last, this technique also justifies the use of source level static analyses, even when an assembly level certification would be required, since it establishes separately that the source and the compiled code are equivalent.

The compilation certification process is performed automatically, thanks to a prover designed specifically. The automatic proof is done at a level of abstraction which has been defined so that the result of the proof of equivalence is strong enough for the goals mentioned above and so that the proof obligations can be solved by efficient algorithms.

The current software features both a C to Power-PC compilation certifier and an interface for an alternate source language frontend, which can be provided by an end-user.

5.6. OpenKappa

Participants: Monte Brown [Harvard Medical School], Vincent Danos [University of Edinburgh], Jérôme Feret [Correspondent], Walter Fontana [Harvard Medical School], Russ Harmer [Harvard Medical School], Jean Krivine [Paris VII].

OPENKAPPA is a collection of tools to build, debug and run models of biological pathways. It contains a compiler for the Kappa Language [61], a static analyzer [60] (for debugging models), a simulator [59], a compression tool for causal traces [58], and a model reduction tool [7],[15],[13].

OPENKAPPA is developped since 2007 and, the OCaml version currently consists of 46 000 lines of OCaml. Software are available in OCaml and in Java. Moreover, an Eclipse pluggin is available.

OPENKAPPA is freely available on the web at http://kappalanguage.org under the LGPL license. Discussion groups are also available on line.

Current external users include the Ecole Polytechnique Federale de Lausanne, the UNAM-Genomics Mexico team. It is used as pedagocical material in graduate lessons at Harvard Medical School, and at the Interdisciplinary Approaches to Life science (AIV) Master Program (Université de Médecine Paris-Descartes).

6. New Results

6.1. Astrée

6.1.1. Application to Aerospace Software

Participants: Julien Bertrane, Patrick Cousot, Radhia Cousot, Jérôme Feret, Laurent Mauborgne, Antoine Miné, Xavier Rival.

In [17], we discuss the principles of static analysis by abstract interpretation and report on the automatic verification of the absence of runtime errors in large embedded aerospace software by static analysis based on abstract interpretation. The first industrial applications concerned synchronous control/command software in open loop. Recent advances consider imperfectly synchronous, parallel programs, and target code validation as well. Future research directions on abstract interpretation are also discussed in the context of aerospace software.

6.1.2. UML

Participants: Julien Bertrane, Patrick Cousot, Radhia Cousot, Jérôme Feret, Laurent Mauborgne, Antoine Miné, Xavier Rival.

Formal methods are increasingly used to help ensuring the correctness of complex, critical embedded software systems. We show how sound semantic static analyses based on abstract interpretation may be used to check properties at various levels of a software design: from high level models to low level binary code. After a short introduction to the Abstract Interpretation theory, we present in [14] a few current applications: checking for run-time errors at the C level, translation validation from C to assembly, and analyzing SAO models of communicating synchronous systems with imperfect clocks. We conclude by briefly proposing some requirements to apply abstract interpretation to modeling languages such as UML.

6.1.3. Industrialization

Participants: Daniel Kästner [AbsInt GmbH], Stephan Wilhelm [AbsInt GmbH], Stephana Nenova [AbsInt GmbH], Patrick Cousot, Radhia Cousot, Jérôme Feret, Laurent Mauborgne, Antoine Miné, Xavier Rival.

Safety-critical embedded software has to satisfy stringent quality requirements. Testing and validation consumes a large and growing fraction of development cost. The last years have seen the emergence of semantics-based static analysis tools in various application areas, from runtime error analysis to worst-case execution time prediction. Their appeal is that they have the potential to reduce testing effort while providing 100% coverage, thus enhancing safety. Static runtime error analysis is applicable to large industry-scale projects and produces a list of definite runtime errors and of potential runtime errors which might be true errors or false alarms. In the past, often only the definite errors were fixed because manually inspecting each alarm was too time-consuming due to a large number of false alarms. Therefore no proof of absence of runtime errors could be given. In [26] the parameterizable static analyzer ASTRÉE is presented. By specialization and parametrization ASTRÉE can be adapted to the software under analysis. This enables ASTRÉE to efficiently compute precise results. ASTRÉE has successfully been used to analyze large-scale safety-critical avionics software with zero false alarms.

6.2. Static Analysis of Parallel Software

Participant: Antoine Miné.

We propose in [17](§ VI) and [28] an analysis of parallel software composed of a fixed set of threads communicating through a shared memory and synchronization primitives (such as mutexes) and scheduled according to strict priorities (so-called real-time scheduling). The analysis is built on top of a generic static analysis of sequential programs with two additional components: a fixpoint iterator to compute and apply an abstraction of thread interferences, and a scheduler partitioning domain to discover and exploit mutual exclusion properties. The analysis has been implemented as the THÉSÉE prototype (see 5.2).

6.3. Numerical Abstractions

Relational abstract domains used to be inefficient when implemented with rationals and unsound when implemented with machine floating point numbers. We have introduced new relational abstract domains which have efficient and sound implementations with floats.

6.3.1. An Abstract Domain to Infer Interval Linear Relationships

Participants: Liqian Chen, Antoine Miné, Ji Wang [National Laboratory for Parallel and Distributed Processing, Changsha, P. R. China], Patrick Cousot.

We introduce in [20] a numerical abstract domain of systems of interval linear equalities, i.e., relations of the form $\sum_k [a_k; b_k]x_k \in [c; d]$. They encompass regular equalities (Karr's domain) as well as some (but not all) inequalities (polyhedra domain) and even some non-convex and unconnected properties (out of the scope of polyhedra). Systems are abstracted into a row echelon form through Gauss elimination algorithms to achieve a polynomial complexity, much lower than classic and interval [46] version of polyhedra. Interval arithmetic on coefficients makes it easy to implement the domain in a sound way using only floating-point arithmetics. The domain was implemented and tested within the Apron framework (see 5.3) with encouraging preliminary results.

6.3.2. Widenings for H-Polyhedra

Participants: Liqian Chen, Axel Simon [Technische Universität München, Garching, Germany].

While the definition of the revised widening for polyhedra is defined in terms of inequalities, most implementations use the double description method as a means to an efficient implementation. In [30], it is shown how standard widening can be implemented in a simple and efficient way using a normalized H-representation (constraint-only) which has become popular in recent approximations to polyhedral analysis. A novel heuristic for this representation is then tuned to capture linear transformations of the state space while ensuring quick convergence for non-linear transformations for which no precise linear invariants exist.

6.3.3. Linear Absolute Value Relation Analysis

Participants: Liqian Chen, Antoine Miné, Ji Wang [National Laboratory for Parallel and Distributed Processing, Changsha, P. R. China], Patrick Cousot.

[21] proposes an abstract domain dealing with linear inequalities involving variables together with their absolute values. It is an extension of the classical linear relation analysis, which permits to deal with some non convex numerical sets. A first nice result states the equivalence between these "linear absolute value inequalities" (AVI) with "interval linear inequalities" (ILI), and "extended linear complementary inequalities" (XLCP, pairs of positive solutions whose pairwise components are not both not zero). The key contribution is the extension of the double-description of polyhedra to XLCP solutions, which is then used to define the standard operations on AVI. The method has been implemented, and experiments show interesting results, with reasonable performances with respect to linear relation analysis.

6.3.4. Graph-Based Weakly Relational Numerical Abstract Domains

Participant: Bouaziz Mehdi.

We introduce a new family of weakly relational numerical abstract domains that achieve improved performances by restricting *a priori* the set of variables related together. It is parametrized by a choice of a graph of variables to relate, and a numerical domain to represent constraints. We focus on variable graphs consisting of complete sub-graphs linked in a tree-structure, for which efficient complete (*i.e.* best precision) algorithms can be constructed. The internship report [39] describes our general construction, and studies in more details the case where the parameter numerical domain is the octagon one.

6.4. Symbolic abstractions

6.4.1. Array Content Analysis

Participants: Patrick Cousot, Radhia Cousot, Francesco Logozzo [Microsoft Research (Redmond, USA)].

In [23], we introduce `FunArray`, a parametric segmentation abstract domain functor for the fully automatic and scalable analysis of array content properties. The functor enables a natural, painless and efficient lifting of existing abstract domains for scalar variables to the analysis of uniform compound data-structures such as arrays and collections (as well as matrices when instantiating the functor on itself). The analysis automatically and semantically divides arrays into consecutive non-overlapping possibly empty segments. Segments are delimited by sets of bound expressions and abstracted uniformly. All bound expressions appearing in a set are equal in the concrete. The `FunArray` can be naturally combined via reduced product with any existing analysis for scalar variables. The bound expressions, the segment abstractions and the reduction operator are the three parameters of the analysis. Once the functor has been instantiated with fixed parameters, the analysis is fully automatic.

We first prototyped `FunArray` in `Arrayal` to adjust and experiment with the abstractions and the algorithms to obtain the appropriate precision/ratio cost. Then we implemented it into CLOUSOT, an abstract interpretation-based static contract checker for `.NET`. We empirically validated the precision and the performance of the analysis by running it on the main libraries of `.NET` and on its own code. We were able to infer thousands of invariants and verify the implementation with a modest overhead (circa 1%). To the best of our knowledge this is the first analysis of this kind applied to such a large code base, and proven to scale.

6.4.2. Segmented Decision Tree Abstract Domain

Participants: Patrick Cousot, Radhia Cousot, Laurent Mauborgne [IMDEA Software (Madrid, Spain)].

The key to precision and scalability in all formal methods for static program analysis and verification is the handling of disjunctions arising in relational analyses, the flow-sensitive traversal of conditionals and loops, the context-sensitive inter-procedural calls, the interleaving of concurrent threads, etc. Explicit case enumeration immediately yields to combinatorial explosion. The art of scalable static analysis is therefore to abstract disjunctions to minimize cost while preserving weak forms of disjunctions for expressivity. Building upon packed binary decision trees to handle disjunction in tests, loops and procedure/function calls and array segmentation to handle disjunctions in array content analysis, we have introduced segmented decision trees in [34] to allow for more expressivity while mastering costs via widenings.

6.4.3. Precondition Inference

Participants: Patrick Cousot, Radhia Cousot, Francesco Logozzo [Microsoft Research (Redmond, USA)].

In the context of program design by contracts, programmers often insert assertions in their code to be optionally checked at runtime, at least during the debugging phase. These assertions would better be given as a precondition of the method/procedure in which they appear. Potential errors would be discovered earlier and, more importantly, the precondition could be used in the context of separate static program analysis as part of the abstract semantics of the code. However in the case of collections (data structures such as arrays, lists, etc) checking both the precondition and the assertions at runtime appears superfluous and costly. So the precondition is often omitted since it is checked anyway at runtime by the assertions. It follows that the static analysis can be much less precise, a fact that can be difficult to understand since "the precondition and assertions are equivalent" (i.e. at runtime, up to the time at which warnings are produced, but not statically) e.g. for separate static analysis.

In [24], we define precisely and formally the contract inference problem from intermittent assertions on scalar variables and elements of collections inserted in the code by the programmer. Our definition excludes no good run even when a non-deterministic choice (e.g. an interactive input) could lead to a bad one. We then introduce new abstract interpretation-based methods to automatically infer both the static contract precondition of a method/procedure and the code to check it at runtime on scalar and collection variables.

6.5. Shape Analysis

We have extended the XISA (eXtensible Inductive Shape Analysis) framework, in order to better deal with low level coding styles and programming languages, and in order to analyze recursive programs in a context dependent way. We also introduced a classification for semantic memory models.

6.5.1. Separating Shape Graphs

Participants: Bor-Yuh Evan Chang [University of Colorado at Boulder (USA)], Vincent Laviron, Xavier Rival.

Detailed memory models that expose individual fields are necessary to precisely analyze code that makes use of low-level aspects such as, pointers to fields and untagged unions. Yet, higher-level representations that collect fields into records are often used because they are typically more convenient and efficient in modeling the program heap. In this [27], we presented a shape graph representation of memory that exposes individual fields while largely retaining the convenience of an object-level model. This representation has a close connection to particular kinds of formulas in separation logic. Then, with this representation, we showed how to extend the Xisa shape analyzer for low-level aspects, including pointers to fields, C-style nested structures and unions, malloc and free, and array values, with minimal changes to the core algorithms (e.g., materialization and summarization).

6.5.2. Concrete Memory Models for Shape Analysis

Participants: Bertrand Jeannet [INRIA Rhône-Alpes, POP'ART Team], Xavier Rival, Pascal Sotin [INRIA Rhône-Alpes, POP'ART Team].

In [31], we discussed four store-based concrete memory models. We characterized memory models by the class of pointers they support and whether they use numerical or symbolic offsets to address values in a block. We gave the semantics of a C-like language within each of these memory models to illustrate their differences. Considering a fragment of Leroy's Clight, including arrays, pointer arithmetics but excluding casts, we linked these concrete memory models with existing shape analyses.

6.5.3. Abstracting Calling-Context with Shapes

Participants: Bor-Yuh Evan Chang [University of Colorado at Boulder (USA)], Xavier Rival.

Interprocedural program analysis is often performed by computing procedure summaries. While possible, computing adequate summaries is difficult, particularly in the presence of recursive procedures. In [29], we propose a complementary framework for interprocedural analysis based on a direct abstraction of the calling context. Specifically, our approach exploits the inductive structure of a calling context by treating it directly as a stack of activation records. We built an abstraction based on separation logic with inductive definitions. A key element of this abstract domain is the use of parameters to refine the meaning of such call stack summaries and thus express relations across activation records and with the heap. In essence, we define an abstract interpretation-based analysis framework for recursive programs that permits a fluid per call site abstraction of the call stack—much like how shape analyzers enable a fluid per program point abstraction of the heap.

6.6. Combining Abstractions

6.6.1. Logical Abstract Domains and Interpretations

Participants: Patrick Cousot, Radhia Cousot, Laurent Mauborgne [IMDEA Software (Madrid, Spain)].

In [33], we give semantic foundations to abstract domains consisting in first order logic formulæ in a theory, as used in verification tools or methods using SMT-solvers or theorem provers. We exhibit conditions for a sound usage of such methods with respect to multi-interpreted semantics and extend their usage to automatic invariant generation by abstract interpretation.

6.6.2. A Framework for Combining Algebraic and Logical Abstract Interpretations

Participants: Patrick Cousot, Radhia Cousot, Laurent Mauborgne [IMDEA Software (Madrid, Spain)].

In [38], [35], we have introduced a reduced product combining algebraic and logical abstractions to design program correctness verifiers and static analyzers by abstract interpretation. The key new idea is to show that the Nelson-Oppen procedure for combining theories in SMT-solvers computes a reduced product in an observational semantics, so that algebraic and logical abstract interpretations can naturally be combined in a classical way using a reduced product on this observational semantics. The main practical benefit is that reductions can be performed within the logical abstract domains, within the algebraic abstract domains, and also between the logical and the algebraic abstract domains, including the case of abstractions evolving during the analysis.

6.6.3. Extrapolation operators for combinations of abstract domains

Participants: Agostino Cortesi [Università Ca'Foscardi di Venizia], Matteo Zanioli.

Extrapolation operators are crucial to ensure the scalability of the analysis to large software systems. In [12], we set the ground for a systematic design of widening and narrowing operators, by comparing the different definitions introduced in the literature and by discussing how to tune them in case of domain abstraction and domains' combination through cartesian and reduced products.

6.7. Abstract Interpretation of Information Flow

Participants: Agostino Cortesi [Università Ca'Foscardi di Venizia], Matteo Zanioli.

The analysis of the flow of information in a program consists in detecting the propagation of sensitive information through the program points of this program. In [22], we propose a new framework that combines variable dependency analysis, based on propositional formulas, and variables' value analysis, based on generic numerical domains.

6.8. Analysis of Biological Pathways

We have introduced a framework to design and analyze biological networks. We focus on protein-protein interaction networks described as graph rewriting systems. Such networks can be used to model some signaling pathways that control the cell cycle. The task is made difficult due to the combinatorial blow up in the number of reachable species (*i.e.*, non-isomorphic connected components of proteins).

6.8.1. Automatic Reduction of Differential Semantics

Participants: Vincent Danos [University of Edinburgh], Jérôme Feret, Walter Fontana [Harvard Medical School], Russ Harmer [Harvard Medical School], Jean Krivine [Paris VII].

We have developed an abstract interpretation-based framework that enables the reduction of the differential semantics for protein-protein interaction networks. Results are sound since trajectories in the abstract system are projections of the trajectories in the concrete system.

This framework has been fully formalized in [15], whereas more conceptual descriptions addressed to a broader audience have been proposed in [7] and [13].

Several talks have also been given on this topic in international workshops [40], and conference [42].

6.8.2. Automatic Reduction of Stochastic Semantics

Participants: Ferdinanda Camporesi, Jérôme Feret, Thomas Henzinger [Institute of Science and Technology, Austria], Heinz Koeppl [École Polytechnique Fédérale de Lausanne], Tatjana Petrov [École Polytechnique Fédérale de Lausanne].

We have proposed an abstract interpretation-based framework for reducing the state-space of stochastic semantics for protein-protein interaction networks.

In [16], we show several examples so as to illustrate why the model reduction that is proposed in [15] for the differential semantics is not sound, in general, for the stochastic semantics.

The model reduction framework for the stochastic semantics is formalized in [25], and the relationships with the notions of lumpability, and bisimulation is established. This framework is explained to a broader audience in [18].

6.8.3. Combining Model Reduction

Participants: Ferdinanda Camporesi, Jérôme Feret, Heinz Koeppl [École Polytechnique Fédérale de Lausanne], Tatjana Petrov [École Polytechnique Fédérale de Lausanne].

In [19], we propose two frameworks for combining model reduction for differential and stochastic semantics.

6.8.4. Rule Refinements

Participants: Vincent Danos [University of Edinburgh], Jérôme Feret, Russ Harmer [Paris VII], Jean Krivine [Harvard Medical School], Elaine Murphy [University of Edinburgh].

We have proposed a formal framework to refine rule-based protein-protein interaction networks while preserving their stochastic and their differential semantics. Refinements is a key process in rule-based modeling. Refining an interaction allows tuning the kinetics of an interaction according to some constraints in the context of the interacting proteins.

In [57], we had proposed a framework to make homogeneous refinements. In such a homogeneous refinement, the accuracy of the refinement is the same for each protein of a given type. In [36], we have extended this framework in order to make heterogeneous refinements, where each agent in a given pattern can be refined independently.

7. Contracts and Grants with Industry

7.1. Asbaprod Contract

ASBAPROD (ASsurance BAsée PRODduit) is an industrial project with Airbus France on static program analysis of synchronous programs by abstract interpretation which objective is determined annually (Sep. 2005–Aug. 2009). The main work in 2010 was on ASTRÉE, as described in Sect. 5.1. Patrick Cousot is the principal investigator for this action.

7.2. Anastasy Contract

ANASTASY (ANAlyse STAtique aSYnchone) is an industrial project with Airbus France on static program analysis of asynchronous programs by abstract interpretation which objective is determined annually (Sep. 2009–Aug. 2012). The main work in 2010 was on THÉSÉE, as described in 5.2. Patrick Cousot is the principal investigator for this action.

7.3. Survol project

SURVOL is an FNRAE (*Fondation de Recherche pour l'Aéronautique et l'Espace*) project (2008–2011) with the University Paul Sabatier Toulouse III and the ENS. The objective is to study automatic verification techniques for embedded robust and stable control software in aerospace still functioning in degraded mode and based on non-smooth optimization and linear matrix inequality (LMI) programming. Patrick Cousot and Radhia Cousot are the principal investigators for this action.

7.4. Ascert project

ASCERT is an FNRAE (*Fondation de Recherche pour l'Aéronautique et l'Espace*) project (2009–2012) between the INRIA-Bretagne Atlantique, the INRIA Rhône-Alpes, the INRIA Paris-Rocquencourt, and the ENS. The goal of this project is to study the use of theorem provers so as to formally certify static analyses. Several approaches are considered: the certification of the static analyzer (once and for all) versus the independent certification of each analyzer run. The project will focus on the certification of some parts of the APRON library and the ASTRÉE analyzer by means of the COQ theorem prover. Patrick Cousot and Radhia Cousot are the principal investigators for this action.

7.5. Sardanes project

SARDANES is an FNRAE (*Fondation de Recherche pour l'Aéronautique et l'Espace*) project (2009–2012) between the University of Perpignan and the ENS. SCADE is widely used to write critical embedded software, as a specification and verification language. The semantics of SCADE uses real arithmetics whereas it is compiled into a language that uses floating-point arithmetics. The goal of the SARDANES project is to use expression transformation so as to ensure that the numerical properties of the programs is preserved during the compilation. Patrick Cousot and Radhia Cousot are the principal investigators for this action.

7.6. Astrée exploitation license agreement

In February 2009 was signed an exploitation license agreement between CNRS, École Normale Supérieure, and AbsInt Angewandte Informatik GmbH for the industrialization of the ASTRÉE analyzer. ASTRÉE is commercially available from AbsInt since January 2010. Continuous work goes on to adapt the ASTRÉE static analyzer to industrial needs, in particular for the automotive industry. Radhia Cousot is the scientific contact.

8. Other Grants and Activities

8.1. AbstractCell ANR

The project ABSTRACTCELL (2009–2013) is sponsored by the ANR-Chair of Excellence program 2009. The overall goal of this project is to investigate formal foundations and computational aspects of both the stochastic and differential approximate semantics for rule-based models. We want to relate these semantics formally, then we want to design sound approximations for each of these semantics (by abstract interpretation) and investigate scalable algorithms to compute the properties of both the stochastic and the differential semantics. Jérôme Feret is the principal investigator for this project.

8.2. European Initiatives

MBAT (Combined model-based analysis and testing of embedded systems) is an ARTEMIS (Advanced Research & Technology for EMbedded Intelligence and Systems, 2011-2014) 7th Framework Programme for Research. MBAT was submitted and accepted in 2010. MBAT will provide Europe with a new leading-edge Reference Technology Platform for effective and cost-reducing validation and verification, focussing primarily on transportation domain, but also to be used in further domains. Developed by European industrial key players (large companies and SMEs) in this domain and supported by leading research partners, this MBAT RTP will be of high value for the European industry, providing very effective means to assure utmost quality embedded systems at reduced costs. With this, MBAT will also strongly support the EU vision of zero traffic fatalities by 2020. As this project is clearly industrial-driven it will be assured that the MBAT RTP will provide solutions for real-life development challenges existing in the European industry as this is also the goal of ARTEMIS projects MBAT involves DAIMLER AG, AIT Austrian Institute of Tech., Aalborg Univ., ABSINT GmbH, Alenia Sia Spa, Advanced Lab. on Embedded Syst., ALSTHOM Transport, AVL List GmbH, BTC Embedded Systems AG, CEA, EADS UK & Deutschland, École Normale Supérieure, Elvior, Enea Services Stockholm AB, FRAUNHOFER, Geensoft, Infineon Tech. Austria AG, IBM SVENSKA AB, MBtech Group GmbH & Co. KGaA, OFFIS E.V, PikeTec GmbH, Prover Technology AB, Ricardo UK Ltd, Rockwell Collins France, Selex Sistemi Integrati, Siemens AG, THALES, Technische Univ. Gratz, Tech. Univ. München, Verified Systems International GmbH, Kompetenzzentrum - Das virtuelle Fahrzeug Forschungsgesellchaft mbH, VOLVO Technolgy AB. The scientific contact is Radhia Cousot.

9. Dissemination

9.1. Interaction with the Scientific Community

9.1.1. Academy Members, Professional Societies

Patrick Cousot is a member of the Academia Europaea.

Patrick Cousot is member of the IFIP working group WG 2.3 on programming methodology.

Patrick Cousot is a member of the Board of Trustees and of the Scientific Advisory Board of the IMDEA-Software (Instituto madrileño de estudios avanzados—Research Institute in Software Development Technology), Madrid, Spain and of the Asian Association for Foundations of Software (AAFS).

Patrick Cousot was a Principal Investigator programme panel member of the Science Foundation Ireland (SFI) (2010).

9.1.2. Collective Responsibilities

Patrick Cousot is director of studies in computer science at ENS and member of the *commission de spécialistes* (hiring committee) of ENS.

Patrick Cousot, Antoine Miné and Xavier Rival are members of the lab council of the Laboratoire d'Informatique de l'École Normale Supérieure.

9.1.3. Editorial Boards and Program Committees

— Patrick Cousot is member of the advisory board of the Higher-Order Symbolic Computation journal (HOSC, Springer) and of the Journal of Computing Science and Engineering (JCSE, Kiise).

Patrick Cousot is member of the steering committees of the Static Analysis Symposium (SAS) and the Verification, Model-Checking and Abstract Interpretation (VMCAI) international conference.

Patrick Cousot was member of the program committees of the 11th International Conference on Verification, Model Checking and Abstract Interpretation (VMCAI 2010), Madrid, Spain, January 17-19, 2010. the 19th European Symposium on Programming (ESOP 2010), Paphos, Cyprus, March 22-26, 2010; the 49th IEEE Conference on Decision and Control Pre-conference Workshop on Verification of Control Systems, Atlanta, GA, USA, December 14, 2010; the 12th International Conference on Verification, Model Checking and Abstract Interpretation (VMCAI 2011), Austin, TX, USA, January 23-25, 2011; and the 14th ACM International Conference on Hybrid Systems (HSCC 2011), Chicago, IL, USA, April 11-14, 2011.

— Radhia Cousot is member of the advisory board of the Higher-Order Symbolic Computation journal (HOSC, Springer) and the Central European Journal of Computer Science (CEJCS, Versita & Springer).

Radhia Cousot is member of the steering committees of the Static Analysis Symposium (SAS), the Workshop on Numerical and Symbolic Abstract Domains (NSAD) and the Workshop on Static Analysis and Systems Biology (SASB).

Radhia Cousot was co-general chair of the 2nd NSAD, 1st SASB, 1st TAPAS, (2010), Perpignan, France and co-program committee chair of the 17th International Static Analysis Symposium (SAS 2010), Perpignan, France, September 14-16, 2010.

Radhia Cousot was member of the program committees of the 31st ACM SIGPLAN 2010 Conference on Programming Language Design and Implementation (PLDI), Toronto, Canada, June 5-10, 2010; the 2nd International Workshop on Abstractions for Petri Nets and Other Models of Concurrency (APNOC 2010), Braga, Portugal, June 22, 2010; the 2nd International Workshop on Numerical and Symbolic Abstract Domains (NSAD 2010), Perpignan, France, 13 September 2010; the 1st International Workshop on Static Analysis and Systems Biology (SASB 2010), Perpignan, France, 13 September 2010; the 1st International Workshop on Tools for Automatic Program AnalysiS (TAPAS 2010), Perpignan, France,17 September 2010; the 21st European Symposium on Programming (ESOP 2011), Saarbrücken, Germany, March 26-April 3, 2011; the 38th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages (POPL 2011), Austin, Texas, USA, January 26-28, 2011.

— Jérôme Feret is a member of the steering committee of the Workshop on Static Analysis and Systems Biology (SASB).

Jérôme Feret was co-program committee chair of the 1st SASB (2010) and is co-program committee chair of the 2nd SASB (2011)

Jérôme Feret was member of the program committee of the 1st International Workshop on Interactions between Computer Science and Biology Workshop (CS2Bio 2010).

— Laurent Mauborgne is member of the steering committees of the Workshop on Numerical and Symbolic Abstract Domains (NSAD).

Laurent Mauborgne was a member of the program committee of the 17th International Static Analysis Symposium (SAS'10).

— Antoine Miné was co-program committee chair of the Second International Workshop on Numerical and Symbolic Abstract Domains (NSAD'10).

Antoine Miné was member of the program committee of the VS-Theory 2010 workshop (part of VSTTE'10). He is member of the program committee of the 18th International Static Analysis Symposium (SAS'11), the third Workshop on Numerical and Symbolic Abstract Domains (NSAD'11), and the First International Workshop on Safety and Security in Cyber-Physical Systems (SSCPS'11).

— Xavier Rival was program committee chair of the workshop on Tools for Automatic Program AnalysiS (TAPAS'10).

Xavier Rival was member of the program committee of the eleventh Conference on Verification, Model Checking, and Abstract Interpretation (VMCAI 2010), and the Conferences on Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2011).

9.1.4. PhD and Habilitation Juries

— Patrick Cousot was a reviewer and in the jury of the PhD thesis of Matthias Péron (September 22, 2010).

Patrick Cousot was a reviewer of the PhD thesis of Tal Lev-Ami (Tel-Aviv University, 2010).

Patrick Cousot was in the jury of the PhD thesis of Tuncay Tekle (Stony Brook University, September 27, 2010) and Christopher Conway (New York University, November 11, 2010).

— Antoine Miné was in the jury of the PhD thesis of Duong Nguyen (École des Mines de Paris, 26 November 2010).

9.2. Teaching

9.2.1. Supervision of PhDs and Internships

— Patrick Cousot and Antoine Miné supervised the research apprenticeship of Liqian Chen.

— Radhia Cousot supervised the PhD thesis of Matteo Zanioli (co-directed thesis with Agostino Cortesi, University of Venice).

— Radhia Cousot and Jérôme Feret supervised the PhD thesis of Ferdinanda Camporesi (co-directed thesis with Maurizio Gabrielli, University of Bologna).

— Antoine Miné supervised the M2 internship of Mehdi Bouaziz (École Normale Supérieure, March–July 2010) [39].

— Xavier Rival supervised the Research internship of Suzanne Renard (September 2010–February 2011).

9.2.2. Graduate Courses

— Patrick Cousot and Radhia Cousot were responsible of the M2 course "Abstract interpretation: application to verification and static analysis" at the MPRI (Master Parisien de Recherche en Informatique). Julien Bertrane, Patrick Cousot, Jérôme Feret, Laurent Mauborgne, Antoine Miné, and Xavier Rival participated in the course.

— Xavier Rival taught the M1 course "INF 563 : Analyse Statique des Programmes" at École Polytechnique. Xavier Rival gave a guest lecture at the University of Colorado at Boulder (Colorado, USA), as part of the Master lecture "CSCI 7135 Program Analysis : Theory and Practice".

9.2.3. Undergraduate Courses

— Julien Bertrane gave practical classes of "Programming Languages and Compilation" at the École Normale Supérieure.

— Xavier Rival gave training sessions on "Principles of Programming Languages" at the École Polytechnique and a lecture on abstract interpretation and static analysis at the École des Mines de Paris.

9.3. PhD theses

Liqian Chen defended his thesis at the National University of Defense Technology at Changsha in China.

9.4. Participation in Conferences and Seminars

9.4.1. Participation in Conferences

VMCAI: International Conference on Verification, Model Checking and Abstract Interpretation (Madrid, Spain, 17-19 January 2010).
Liqian Chen, Antoine Miné, Xavier Rival and Matteo Zanioli attended the conference. Xavier Rival chaired a session. Liqian Chen gave a talk [45].

POPL: ACM Symposium on Principles of Programming Languages (Madrid, Spain, 20-22 January 2010).
Antoine Miné, Xavier Rival and Matteo Zanioli attended the conference.

CMBS: Workshop Computational Modelling of Biological Systems (Holetown, Barbados, 15-19 March 2010). Jérôme Feret attended the workshop and gave a three hours talk on static analysis of biological pathways [43] and a two hours talk on model reduction of differential models [40].

ESOP: European Symposium on Programming (Paphos, Cyprus), 22–26 March 2010).
Vincent Laviron and Xavier Rival attended the conference. Vincent Laviron presented [27].

MFPS: International Conference on Mathematical Foundations of Programming Semantics (Ottawa, Ontario,Canada, 6–20 May 2010).
Jérôme Feret attended the conference and gave a talk on model reduction of differential models [42].

FMOODS: International Conference on Formal Methods for Open Object-based Distributed Systems (Amsterdam, Netherlands, 6–11 June 2010).
Ferdinanda Camporesi and Jérôme Feret attended the conference.

CS2Bio: International Workshop on Interactions between Computer Science and Biology (Amsterdam, Netherlands, 12 June 2010).
Ferdinanda Camporesi and Jérôme Feret attended the workshop. Jérôme Feret gave an invited talk on model reduction of differential models [41].

LSB: Workshop on Logic and Systems Biology (Edinburgh, Scotland, UK, 10 July 2010).
Ferdinanda Camporesi and Jérôme Feret attended the workshop.

LICS: IEEE Symposium on Logic In Computer Science (Edinburgh, Scotland, UK, 11-14 July 2010).
Ferdinanda Camporesi and Jérôme Feret attended the conference.

CAV: International Conference on Computer Aided Verification (Edinburgh, Scotland, UK, 15–19 July 2010).
Ferdinanda Camporesi attended the conference.

MecBIC: International Workshop on Membrane Computing and Biologically Inspired Process Calculi (Iena, Germany, 23 August 2010).
Jérôme Feret attended the workshop and gave a talk [25].

SOS: International Workshop on Structural Operational Semantics (Paris, France, 30 August 2010).
Ferdinanda Camporesi attended the workshop.

CONCUR: International Conference on Concurrency Theory (Paris, France, 31 August – 3 September 2010).
Ferdinanda Camporesi and Jérôme Feret attended the conference.

SASB: International Workshop on Static Analyis and Systems Biology (Perpignan, France, 13 September 2010).
Ferdinanda Camporesi, Patrick Cousot, Radhia Cousot, and Jérôme Feret attended to the workshop. Jérôme Feret co-chaired the workshop and chaired a session.

NSAD: Second International Workshop on Numerical and Symbolic Abstract Domains.
Antoine Miné, Mehdi Bouaziz and Xavier Rival attended the workshop. Antoine Miné co-chaired the workshop.

SAS: 17th International Static Analysis Symposium (Perpignan, France, 14–16 September 2010).
Ferdinanda Camporesi, Patrick Cousot, Radhia Cousot, Jérôme Feret, Laurent Mauborgne, Antoine Miné, Xavier Rival, Mehdi Bouaziz, and Matteo Zanioli attended the conference.

TAPAS: International Workshop on Tools for Automatic Program Analysis (Perpignan, France, 17 September).
Ferdinanda Camporesi, Patrick Cousot, Radhia Cousot, Jérôme Feret, Laurent Mauborgne, Antoine Miné, Xavier Rival, Mehdi Bouaziz attended the workshop. Xavier Rival co-chaired the workshop and chaired a session.

ICNAAM: International Conference of Numerical Analysis and Applied Mathematics (Rhodos, Grece, 19–25 September 2010).
Jérôme Feret attended the workshop and gave a talk [16].

UML&FM: Third IEEE International workshop UML and Formal Methods (Shanghai, China, 16 November 2010).
Julien Bertrane attended the workshop and gave a talk [14].

PLDI: ACM SIGPLAN 2010 Conference on Programming Language Design and Implementation (Toronto, Canada, 7–9 June 2010).
Antoine Miné and Laurent Mauborgne attended the conference.

LFX: First International Workshop on Learning From eXperience (Toronto, Canada, 6 June 2010).
Antoine Miné and Laurent Mauborgne attended the workshop, and Antoine Miné gave a talk [44].

9.4.2. Invitations and Participation in Seminars

— Patrick Cousot gave a talk on *Abstract interpretation: from origin to perspectives* at the Dagstuhl Perspectives Workshop 10482 on "Formal Methods — Just a Euro-Science?", Schloss Dagstuhl, Germany, 30 November 2010 – 3 December 2010; on *Contract Precondition Inference from Intermittent Assertions on Collections*, at the The Future of Software Engineering (FOSE) Symposium, ETH Zürich, Switzerland, 22-23 November 2010 [24]; on *Static Analysis and Verification of Aerospace Software by Abstract Interpretation* at the Workshop on formal verification of avionics software products, Airbus France, Toulouse, France, June 24, 2010; on *A Scalable Segmented Decision Tree Abstract Domain* at the Amir Pnueli Memorial Symposium, CIMS, NYU, New York, NY, USA, May 7—9, 2010 [34]; on *Embedded software verification by abstract interpretation* at a visit to Rockwell-Collins, Cedar Rapids, Iowa, US, May 3rd, 2010; on *Array content static analysis by segmentation* at a Seminar of IBM T.J. Watson Research Center at Hawthorn, NY, USA, April 24, 2010 [23]; on *Challenge Problems in Aerospace Software Verification*, at Carnegie Mellon University, Pittsburgh, PA, USA. March 4th, 2010 [17].

— Patrick Cousot and Radhia Cousot gave a talk on *Contract Precondition Inference from Intermittent Assertions on Collections* at the Software Engineering and Programming Languages Seminar, MSR, Redmond, WA, USA, September 8, 2010 [24].

— Jérôme Feret presented an abstract interpretation framework for reducing differential semantics for biological networks in the Bio Systems analysis group at the university of Jena and in the Logic and semantics seminar at the university of Cambridge.

— Antoine Miné gave a talk on the Apron library at the seminar of the "Équipe Contraintes, LINA," École des Mines de Nantes, France, 10 February 2010.

— Xavier Rival was invited to give a talk on “Shape Analysis with Inductive Definitions” at Liafa in Paris in March 2010. Xavier Rival was invited to give a talk on “Shape Analysis with Inductive Definitions” at IMDEA-Software in Madrid in april 2010. Xavier Rival was invited to give a talk on abstract interpretation based static analysis at the University of Colorado at Boulder. Xavier Rival gave a talk on the Abstraction of Calling Contexts with Shapes at the University of Queen Mary (London, UK) in october 2010. Xavier Rival was invited to give a talk on “Shape Analysis and Interprocedural Analysis” at the Workshop on “Automata and Logic for Data Manipulating programs” (Paris, December 2010).

— Laurent Mauborgne gave a talk on “Modification Shapes”, and another talk on “Segmented Relations” at IMDEA Software in Madrid in March 2010.

10. Bibliography

Major publications by the team in recent years

[1] J. BERTRANE, P. COUSOT, R. COUSOT, J. FERET, L. MAUBORGNE, A. MINÉ, X. RIVAL. *Static Analysis and Verification of Aerospace Software by Abstract Interpretation*, in "Proceedings of the American Institue of Aeronautics and Astronautics (AIAA Infotech@Aerospace 2010)", Atlanta, Georgia, USA, American Institue of Aeronautics and Astronautics, 2010, http://hal.inria.fr/inria-00528611.

[2] B. BLANCHET, P. COUSOT, R. COUSOT, J. FERET, L. MAUBORGNE, A. MINÉ, D. MONNIAUX, X. RIVAL. *A Static Analyzer for Large Safety-Critical Software*, in "Proceedings of the ACM SIGPLAN 2003 Conference on Programming Language Design and Implementation (PLDI'03)", ACM Press, June 7–14 2003, p. 196–207.

[3] P. COUSOT. *Constructive Design of a Hierarchy of Semantics of a Transition System by Abstract Interpretation*, in "Theoretical Computer Science", 2002, vol. 277, n^{o} 1–2, p. 47–103.

[4] P. COUSOT, R. COUSOT. *Temporal Abstract Interpretation*, in "Conference Record of the Twentyseventh Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages", ACM Press, New York, United States, January 2000, p. 12–25.

[5] P. COUSOT, R. COUSOT. *Systematic Design of Program Transformation Frameworks by Abstract Interpretation*, in "Conference Record of the Twentyninth Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages", ACM Press, New York, United States, January 2002, p. 178–190.

[6] P. COUSOT, R. COUSOT. *Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints*, in "Conference Record of the Fourth Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages", ACM Press, New York, United States, 1977, p. 238–252.

[7] J. FERET, V. DANOS, J. KRIVINE, R. HARMER, W. FONTANA. *Internal coarse-graining of molecular systems*, in "Proceeding of the national academy of sciences", Apr 2009, vol. 106, n^{o} 16, http://hal.inria.fr/inria-00528330.

[8] L. MAUBORGNE, X. RIVAL. *Trace Partitioning in Abstract Interpretation Based Static Analyzers*, in "European Symposium on Programming (ESOP'05)", M. SAGIV (editor), Lecture Notes in Computer Science, Springer-Verlag, 2005, vol. 3444, p. 5–20.

[9] A. MINÉ. *The Octagon Abstract Domain*, in "Higher-Order and Symbolic Computation", 2006, vol. 19, p. 31–100.

[10] X. RIVAL. *Symbolic Transfer Functions-based Approaches to Certified Compilation*, in "Conference Record of the Thirtyfirst Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages", ACM Press, New York, United States, 2004, p. 1–13.

Publications of the year

Doctoral Dissertations and Habilitation Theses

[11] L. CHEN. *Sound floating-point and non-convex static analysis using interval linear abstract domains*, National University of Defense Technology, Changsha (P.R. China), April 2010.

Articles in International Peer-Reviewed Journal

[12] A. CORTESI, M. ZANIOLI. *Widening and narrowing operators for abstract interpretation*, in "Computer Languages, Systems and Structures", 2011, vol. 37, n^{o} 1, p. 24 - 42, http://dx.doi.org/10.1016/j.cl.2010.09.001.

[13] R. HARMER, V. DANOS, J. FERET, J. KRIVINE, W. FONTANA. *Intrinsic Information carriers in combinatorial dynamical systems*, in "Chaos", 2010, vol. 20, n^{o} 3, 037108, http://hal.inria.fr/hal-00520128.

Invited Conferences

[14] J. BERTRANE, P. COUSOT, R. COUSOT, J. FERET, L. MAUBORGNE, A. MINÉ, X. RIVAL. *Static Analysis by Abstract Interpretation of Embedded Critical Software*, in "Proceedings of the Third IEEE International workshop UML and Formal Methods", Shangai, China, A. CANAL (editor), November 16th 2010, http://hal.inria.fr/inria-00528632.

[15] V. DANOS, J. FERET, W. FONTANA, R. HARMER, J. KRIVINE. *Abstracting the differential semantics of rule-based models: exact and automated model reduction*, in "Proceedings of Logic in Computer Science (LICS 2010)", Edinburgh, UK, J.-P. JOUANNAUD (editor), 2010, p. 362–381, http://hal.inria.fr/hal-00520112.

[16] J. FERET. *Fragments-based model reduction: some case studies*, in "Proceedings of the First International Workshop on Interactions between Computer Science and Biology (CS2Bio 2010)", Amsterdam, The Netherlands, J. KRIVINE, A. TROINA (editors), Elsevier, 2010, http://hal.inria.fr/inria-00527960.

International Peer-Reviewed Conference/Proceedings

[17] J. BERTRANE, P. COUSOT, R. COUSOT, J. FERET, L. MAUBORGNE, A. MINÉ, X. RIVAL. *Static Analysis and Verification of Aerospace Software by Abstract Interpretation*, in "Proceedings of the American Institue of Aeronautics and Astronautics (AIAA Infotech@Aerospace 2010)", Atlanta, Georgia, USA, American Institue of Aeronautics and Astronautics, 2010, http://hal.inria.fr/inria-00528611.

[18] F. CAMPORESI, J. FERET, H. KOEPPL, T. PETROV. *Automatic reduction of stochastic rules-based models in a nutshell*, in "Proceedings of the International Conference of Numerical Analysis and Applied Mathematics (ICNAAM 2010)", Rhodos, Greece, American Institute of Physics, Sep 2010, vol. 1281(2), p. 1330-1334, http://hal.inria.fr/inria-00527548.

[19] F. CAMPORESI, J. FERET, H. KOEPPL, T. PETROV. *Combining Model Reductions*, in "Proceedings of the 26th Conference on the Mathematical Foundations of Programming Semantics (MFPS 2010)", Canada Ottawa, M. MISLOVE, P. SELINGER (editors), Elsevier, Sep 2010, vol. 265, p. 73–96, http://hal.inria.fr/inria-00527536.

[20] L. CHEN, A. MINÉ, J. WANG, P. COUSOT. *An Abstract Domain to Discover Interval Linear Equalities*, in "Proceedings of the 11th International Conference on Verification, Model Checking, and Abstract Interpretation (VMCAI'10)", Madrid, Spain, Lecture Notes in Computer Science, Springer, January 2010, vol. 5944, p. 112-128, http://hal.archives-ouvertes.fr/hal-00531563/en/.

[21] L. CHEN, A. MINÉ, J. WANG, P. COUSOT. *Linear Absolute Value Relation Analysis*, in "Proceedings of the 20th European Symposium on Programming (ESOP 2011)", Saarbrücken, Germany, Lecture Notes in Computer Science, Springer, 2010, to appear.

[22] A. CORTESI, M. ZANIOLI. *Information Leakage Analysis by Abstract Interpretation*, in "the 37th International Conference on Current Trends in Theory and Practice of Computer Science", Novy Smokovec Slovakia, Lecture Notes in Computer Science, Springer, 2011, to appear.

[23] P. COUSOT, R. COUSOT, F. LOGOZZO. *A Parametric Segmentation Functor for Fully Automatic and Scalable Array Content Analysis*, in "Proceedings of the 38 th Annual ACM Symposium on Principles Of Programming Languages (POPL'2011)", Austin, Texas, ACM Press, January 26–28 2011, to appear, http://hal.inria.fr/inria-00543874/en/.

[24] P. COUSOT, R. COUSOT, F. LOGOZZO. *Contract Precondition Inference from Intermittent Assertions on Collections*, in "Proceedings of the 12th Conference on Verification, Model Checking and Abstract Interpretation (VMCAI'11)", Springer, January 2011, (to appear), http://hal.inria.fr/inria-00543881/en/.

[25] J. FERET, T. HENZINGER, H. KOEPPL, T. PETROV. *Lumpability Abstractions of Rule-based Systems*, in "Proceedings of the 4th Workshop on Membrane Computing and Biologically Inspired Process Calculi (MeCBIC0 2010", Jena, Germany, G. CIOBANU, M. KOUTNY (editors), Elsevier, 2010, http://hal.inria.fr/inria-00527971.

[26] D. KÄSTNER, S. WILHELM, S. NENOVA, P. COUSOT, R. COUSOT, J. FERET, L. MAUBORGNE, A. MINÉ, X. RIVAL. *Astree: Proving the Absence of Runtime Errors*, in "Proceedings of the Embedded real time software and systems (ERTS2 2010)", Toulouse , France, AAAF, SEE, SIA, 2010, http://hal.inria.fr/inria-00528600.

[27] V. LAVIRON, B.-Y. E. CHANG, X. RIVAL. *Separating Shape Graphs*, in "Proceedings of the 19th European Symposium on Programming (ESOP 2010)", Paphos, Chypre, A. GORDON (editor), Lecture Notes in Computer Science, Springer, March 2010, vol. 6012, p. 387–406, http://hal.inria.fr/inria-00539548/en/.

[28] A. MINÉ. *Static Analysis of Run-Time Errors in Embedded Critical Parallel C Programs*, in "Proceedings of the 20th European Symposium on Programming (ESOP 2011)", Saarbrücken, Germany, Lecture Notes in Computer Science, Springer, 2010, to appear.

[29] X. RIVAL, B.-Y. E. CHANG. *Calling Contexts Abstraction with Shapes*, in "Proceedings of the 38 th Annual ACM Symposium on Principles Of Programming Languages (POPL'2010)", Austin, Texas, ACM Press, January 26–28, 2011, (to appear).

[30] A. SIMON, L. CHEN. *Simple and Precise Widenings for H-Polyhedra*, in "8th Asian Symposium on Programming Languages and Systems (APLAS 2010)", Shanghai, China, Lecture Notes in Computer Science, Springer, Nov 28–Dec 1, 2010, vol. 6461, p. 139–155.

[31] P. SOTIN, B. JEANNET, X. RIVAL. *Concrete Memory Models for Shape Analysis*, in "Proceedings of the Second International Workshop onNumerical And Symbolic Abstract Domains (NSAD 2010)", Perpignan, France, September 2010, http://hal.inria.fr/inria-00539553/en/.

Scientific Books (or Scientific Book chapters)

[32] P. COUSOT, R. COUSOT. *A gentle introduction to formal verification of computer systems by abstract interpretation*, in "Logics and Languages for Reliability and Security", J. ESPARZA, O. GRUMBERG, M. BROY (editors), NATO Science Series III: Computer and Systems Sciences, IOS Press, 2010, p. 1–29, http://hal.inria.fr/inria-00543886/en/.

[33] P. COUSOT, R. COUSOT, L. MAUBORGNE. *Logical Abstract Domains and Interpretations*, in "The Future of Software Engineering", S. NANZ (editor), Springer, 2010, p. 48–71, http://hal.inria.fr/inria-00543855/en/.

[34] P. COUSOT, R. COUSOT, L. MAUBORGNE. *A Scalable Segmented Decision Tree Abstract Domain*, in "Time for Verification, Essays in Memory of Amir Pnueli", Z. MANNA, D. A. PELED (editors), Lecture Notes in Computer Science, Springer, May 2010, vol. 6200, p. 72–95, http://hal.inria.fr/inria-00543632/en/.

[35] P. COUSOT, R. COUSOT, L. MAUBORGNE. *The reduced product of abstract domains and the combination of decision procedures*, in "14th International Conference on Fondations of Software Science and Computation Structures (FoSSaCS 2011), Saarbrücken, Germany", M. HOFMANN (editor), Lecture Notes in Computer Science, Springer, March 26 – April 3, 2011, to appear.

[36] E. MURPHY, V. DANOS, J. FERET, J. KRIVINE, R. HARMER. *Rule Based Modeling and Model Refinement*, in "Elements of Computational Systems Biology", H. LODHI, S. MUGGLETON (editors), John Wiley & Sons, 2010, p. 83–114, http://hal.inria.fr/inria-00527981.

Books or Proceedings Editing

[37] R. COUSOT, M. MARTEL (editors). *Static Analysis – 17th International Symposium, SAS 2010, Perpignan, France, September 14-16, 2010. Proceedings*, Lecture Notes in Computer Science, Springer, 2010, vol. 6337.

Research Reports

[38] P. COUSOT, R. COUSOT, L. MAUBORGNE. *A Framework for Combining Algebraic and Logical Abstract Interpretations*, CNRS, ENS, INRIA, IMDEA-Software, July 16, 2010, http://hal.inria.fr/inria-00543890/en/.

Other Publications

[39] M. BOUAZIZ. *TreeKs: un foncteur pour faire passer les domaines numériques à l'échelle*, École Normale Supérieure, Paris, France, August 2010.

[40] J. FERET. *Automatic reduction of differential semantics for protein-protein interaction networks, by abstract interpretation*, 19 March 2010, International Workshop on Computational Modelling of Biological Systems, Holetown, Barbados.

[41] J. FERET. *Internal coarse-graining of molecular systems*, 12 June 2010, Invited talk at the First International Workshop on Interactions between Computer Science and Biology, Amsterdam, Netherland.

[42] J. FERET. *Internal coarse-graining of molecular systems*, 8 May 2010, MFPS XXVI, Special Session on Systems Biology.

[43] J. FERET. *Reachability Analysis of Signalling Pathways, by Abstract Interpretation*, 15 March 2010, International Workshop on Computational Modelling of Biological Systems, Holetown, Barbados.

[44] A. MINÉ. *Astrée: Building a Static Analyzer for Real-Life Programs*, 6 June 2010, invited talk at the First International Workshop on Learning From eXperience, Toronto, Canada.

References in notes

[45] L. CHEN, A. MINÉ, P. COUSOT. *A Sound Floating-Point Polyhedra Abstract Domain*, in "Proc. of the Sixth Asian Symposium on Programming Languages and Systems (APLAS'08)", Lecture Notes in Computer Science, Springer, December 2008, vol. 5356, p. 3–18.

[46] L. CHEN, A. MINÉ, J. WANG, P. COUSOT. *Interval Polyhedra: An Abstract Domain to Infer Interval Linear Relationships*, in "Proceedings of the 16th International Static Analysis Symposium (SAS'09)", LNCS, Springer, August 2009, vol. 5673, p. 309–325.

[47] P. COUSOT. *Proving the Absence of Run-Time Errors in Safety-Critical Avionics Code, invited tutorial*, in "Proceedings of the Seventh ACM & IEEE International Conference on Embedded Software, EMSOFT'2007", C. M. KIRSCH, R. WILHELM (editors), ACM Press, New York, USA, 2007, p. 7–9.

[48] P. COUSOT. *Méthodes itératives de construction et d'approximation de points fixes d'opérateurs monotones sur un treillis, analyse sémantique de programmes (in French)*, Université scientifique et médicale de Grenoble, Grenoble, France, 21 March 1978.

[49] P. COUSOT. *The Calculational Design of a Generic Abstract Interpreter, invited chapter*, in "Calculational System Design", M. BROY, R. STEINBRÜGGEN (editors), NATO Science Series, Series F: Computer and Systems Sciences. IOS Press, Amsterdam, The Netherlands, 1999, vol. 173, p. 421–505.

[50] P. COUSOT, R. COUSOT. *Basic Concepts of Abstract Interpretation, invited chapter*, in "Building the Information Society", R. JACQUART (editor), Kluwer Academic Publishers, Dordrecht, The Netherlands, 2004, chap. 4, p. 359–366.

[51] P. COUSOT, R. COUSOT. *Grammar Analysis and Parsing by Abstract Interpretation, invited chapter*, in "Program Analysis and Compilation, Theory and Practice: Essays dedicated to Reinhard Wilhelm on the Occasion of his 60th Birthday", T. W. REPS, M. SAGIV, J. BAUER (editors), Lecture Notes in Computer Science, Springer, Berlin, Germany, 2007, vol. 4444.

[52] P. COUSOT, R. COUSOT. *Bi-inductive structural semantics*, in "Information and Computation", 2009, vol. 207, n^{o} 2, p. 258–283.

[53] P. COUSOT, R. COUSOT. *Systematic design of program analysis frameworks*, in "Conference Record of the Sixth Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages", ACM Press, New York, United States, 1979, p. 269–282, http://www.di.ens.fr/~cousot/COUSOTpapers/POPL79.shtml.

[54] P. COUSOT, R. COUSOT, J. FERET, L. MAUBORGNE, A. MINÉ, D. MONNIAUX, X. RIVAL. *The* ASTRÉE *analyser*, in "Proceedings of the Fourteenth European Symposium on Programming Languages and Systems,

ESOP'2005, Edinburg, Scotland", M. SAGIV (editor), Lecture Notes in Computer Science, Springer, Berlin, Germany, 2–10 April 2005, vol. 3444, p. 21–30.

[55] P. COUSOT, R. COUSOT, J. FERET, L. MAUBORGNE, A. MINÉ, D. MONNIAUX, X. RIVAL. *Varieties of Static Analyzers: A Comparison with* ASTRÉE*, invited paper*, in "Proceedings of the First IEEE & IFIP International Symposium on Theoretical Aspects of Software Engineering, TASE'07", Shanghai, China, M. HINCHEY, J. HE, J. SANDERS (editors), IEEE Computer Society Press, Los Alamitos, California, USA, 6–8 June 2007.

[56] P. COUSOT, R. COUSOT, R. GIACOBAZZI. *Abstract Interpretation of Resolution-Based Semantics*, in "Theoretical Computer Science", Nov. 2009, vol. 410, n^{o} 46.

[57] V. DANOS, J. FERET, W. FONTANA, R. HARMER, J. KRIVINE. *Rule-based modelling, symmetries, refinements.*, in "Proceedings of the First International Workshop, Formal Methods in Systems Biology, FMSB'2008", Cambridge, UK, J. FISHER (editor), Lecture Notes in BioInformatics, Springer, Berlin, Germany, 4–5 June 2008, vol. 5054, p. 103–122.

[58] V. DANOS, J. FERET, W. FONTANA, R. HARMER, J. KRIVINE. *Rule-based modelling of cellular signalling*, in "International Conference on Concurrency Theory (CONCUR'07)", Portugal, September 2007, http://hal.archives-ouvertes.fr/hal-00164297/en/.

[59] V. DANOS, J. FERET, W. FONTANA, J. KRIVINE. *Scalable Simulation of Cellular Signaling Networks*, in "the 5th Asian Symposium on Programming Languages and Systems - APLAS'07", Z. SHAO (editor), Lecture Notes in Computer Science, Springer, 2007, vol. 4807, p. 139-157 [*DOI :* 10.1.1.139.5120], http://hal.inria.fr/inria-00528409/en/.

[60] V. DANOS, J. FERET, W. FONTANA, J. KRIVINE. *Abstract Interpretation of Cellular Signalling Networks*, in "the 9th International Conference on Verification, Model Checking and Abstract Interpretation - VMCAI'08", F. LOGOZZO, D. A. PELED, L. D. ZUCK (editors), Lecture Notes in Computer Science, Springer, 2008, vol. 4905, p. 83-97 [*DOI :* 10.1007/978-3-540-78163-9_11], http://hal.inria.fr/inria-00528352/en/.

[61] V. DANOS, C. LANEVE. *Formal Molecular Biology*, in "Theoretical Computer Science", 10 2004, vol. 325, n^{o} 1, p. 69-110 [*DOI :* 10.1016/J.TCS.2004.03.065], http://hal.archives-ouvertes.fr/hal-00164591/en/.