INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Project-Team Abstraction

Abstract Interpretation

Paris - Rocquencourt

THEME ALGORITHMICS, PROGRAMMING, SOFTWARE AND ARCHITECTURE

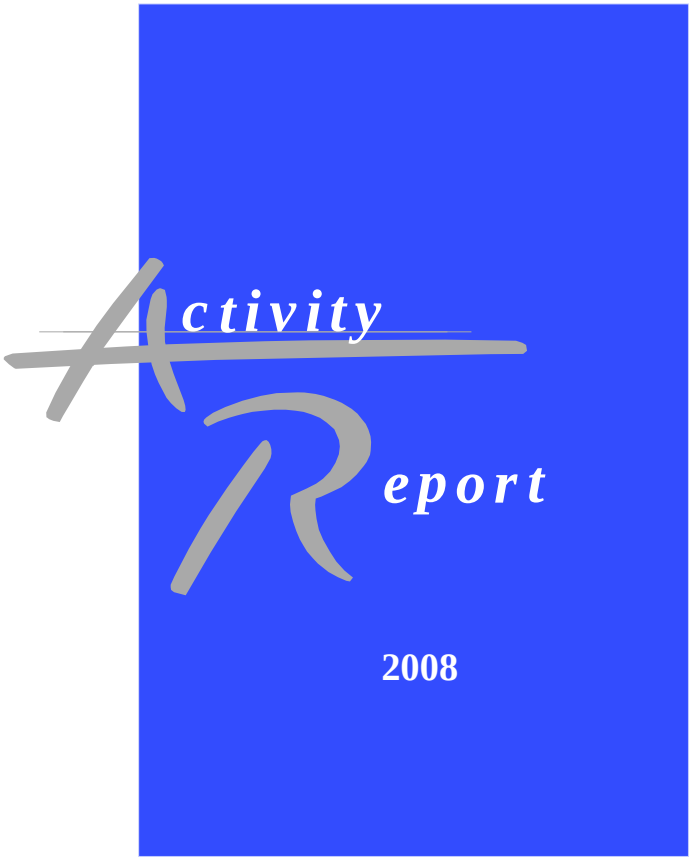

Table of contents

1. Team

Research Scientist

Bruno Blanchet [CR, CNRS, HdR]
Radhia Cousot [DR, CNRS, HdR]
Jérôme Feret [CR, INRIA Paris–Rocquencourt]
Antoine Miné [CR, CNRS]
Xavier Rival [CR, INRIA Paris–Rocquencourt]

Faculty Member

Patrick Cousot [Team leader, Professor/Professeur, ENS, HdR]
Laurent Mauborgne [Assistant Professor/Maître de conférences, ENS, HdR]

Technical Staff

Élodie-Jane Sims [Research engineer, ENS]

PhD Student

Julien Bertrane
Guillaume Capron
Pietro Ferrara

Post-Doctoral Fellow

Axel Simon [University of Kent]

Visiting Scientist

Roberto Giacobazzi [Università di Verona, June — Aug. 2008]

Administrative Assistant

Joëlle Isnard [Administrative Head DI, ENS]
Elisabeth Baque [INRIA, 16 Oct. 2008 —]
Nathalie Gaudechoux [INRIA, — 23 March. 2008]
Emmanuelle Grousset [INRIA, 13 March. 2008 — 15 Oct. 2008]
Nelly Maloisel [INRIA, 24 March. 2008 —]

Other

Ferdinanda Camporesi [Università di Bologna, — 31 March 2008]
Liqian Chen [National University of Defense Technology, — 30 Sept. 2008]
David Durrleman [ENS, —30 Jan. 2008]

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 and development 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.

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 of the Year

In 2008, the abstract interpretation community celebrated the *30 years of Abstract Interpretation* during a one day workshop in San Francisco, USA. Fourteen speakers from eleven countries gave a broad overview of the global spread of abstract interpretation worldwide during the last 30 years. The publication of the English translation of Patrick Cousot's Thesis is ongoing.

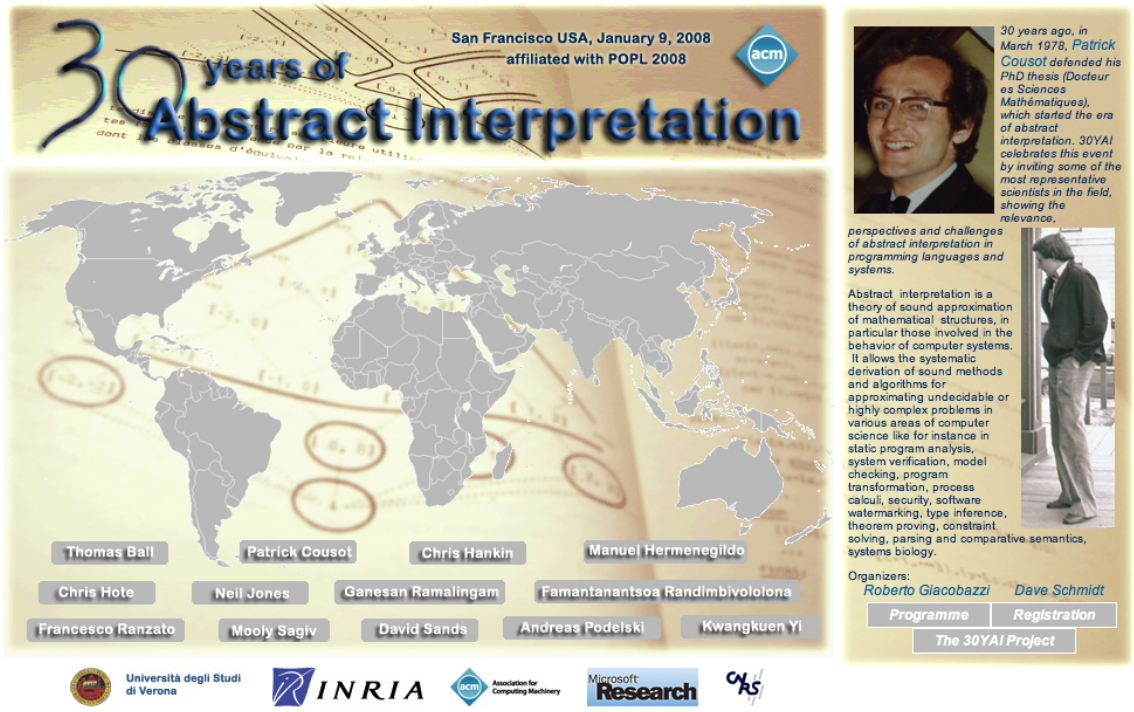

Figure 1.

Bruno Blanchet defended his *habilitation à diriger des recherches* on November 26. His report [12] summarizes his work on the verification of security protocols, which essentially consists in designing and implementing the two protocol verifiers ProVerif (Section 5.4) and CryptoVerif (Section 5.5). ProVerif is already widely used. CryptoVerif is more recent and starts being used by others; it is the first automatic protocol verifier that is sound in the computational model.

Patrick Cousot has received a *Humboldt Research Award* 2008.

3. Scientific Foundations

3.1. Abstract Interpretation Theory

The abstract interpretation theory [80], [6], [85] 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.

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, and so, automatic tools for reasoning about programs are all either unsound or incomplete, or they may not terminate, or they require human intervention. 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. [83], [84]) or incomplete (e.g. [2]).

Sound, automatic, terminating and precise tools are difficult to design. Complete tools to solve non-trivial verification problems are impossible to design, 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 [86]. 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], [85];
- guiding the correct formal design of automatic tools for *program analysis* (computing an abstract semantics) and *program verification* (proving that an abstract semantics satisfies an abstract specification) [81].

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 seven years, aims at proving the absence of runtime errors in programs written in the C programming language. It is used in the avionics industry to verify very large, synchronous, time-triggered, real-time, safety-critical, embedded software.

3.3. Advanced Introductions to Abstract Interpretation

The informal presentation "Abstract Interpretation in a Nutshell" aims at providing a short intuitive introduction to the theory. A more comprehensive introduction to abstract interpretation is available online[1]. The paper entitled "Basic concepts of abstract interpretation" [82] and an elementary "course on abstract interpretation"[2] can also be found on the web.

4. Application Domains

4.1. Certification of Safety Critical Software

Keywords: *absence of runtime error*, *abstract interpretation*, *certified compilation*, *static analysis*, *translation validation*, *verifier*.

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.

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, sound program analysis methods such as abstract-interpretation-based static analysis are able to cope with these programs, since they can prove the absence of bugs. Yet, these techniques are generally incomplete since the absence of runtime errors is undecidable in practice; therefore, they are prone to false alarms (*i.e.*, they may fail to prove the absence of runtime errors for a program which is safe).

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 (hence, producing a full proof of absence of runtime error) 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 [79], [87], which has been initially designed specifically for synchronous embedded software.

[1] http://www.di.ens.fr/~cousot/AI/
[2] http://web.mit.edu/afs/athena.mit.edu/course/16/16.399/www/

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 for seven years, using abstract interpretation techniques. Successful results have been achieved on industrial applications. The ABSTRACTION project-team has strong plans to continue research on this topic and to industrialize ASTRÉE, which should be commercially available in the near future.

4.2. Security Protocols

Keywords: *formal model*, *security protocols*, *verifier*.

Security protocols use cryptography in order to guarantee the security of exchanges over an insecure network, such as the Internet. The design of security protocols is notoriously error-prone: errors have been found in many published protocols. Security errors can have serious consequences, such as loss of money in the case of electronic commerce. Moreover, security errors cannot be detected by testing, because they appear only in the presence of a malicious adversary. Security protocols are therefore an important area for formal verification.

The work of the ABSTRACTION project-team on security protocols has led to the development of two successful automatic protocol verifiers, PROVERIF in the formal model and CRYPTOVERIF in the computational model, and we plan to pursue research on this topic, in particular with extensions to CRYPTOVERIF.

4.3. Abstraction of Biological Cell Signalling Networks

Keywords: *biology*, *health*, *static analysis*.

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 modelling 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 signalling networks ranges from qualitative abstraction to quantitative abstraction.

5. Software

5.1. The Astrée Static Analyzer

Keywords: *absence of runtime error*, *abstract interpretation*, *static analysis*, *verifier*.

Participants: Patrick Cousot [project leader, correspondant], Radhia Cousot, Jérôme Feret, Laurent Mauborgne, Antoine Miné, Xavier Rival, Bruno Blanchet [Nov. 2001–Nov. 2003], David Monniaux [Nov. 2001–Aug. 2007].

The ASTRÉE static analyzer [79], [87] 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.

5.2. The Apron Numerical Abstract Domain Library

Keywords: *convex polyhedron*, *interval*, *linear equality numerical abstract domain*, *octagon*.

Participants: Antoine Miné [correspondant], 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, 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, richer, API, with C, C++, and OCaml bindings.

The APRON library is freely available on the web at http://apron.cri.ensmp.fr/library under the LGPL license. In order to help disseminate the knowledge in 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 86 000 lines of C, C++, and OCaml. This year has seen the release of version 0.9.9. The main addition is the support for arbitrary precision floats through the MPFR library. This enables floating-point capable domains (such as intervals and octagons) to use arbitrary precision, and adds the MPFR data-types to the API and support libraries.

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).

5.3. Translation Validation

Keywords: *abstract interpretation*, *certified compilation*, *static analysis*, *translation validation*, *verifier*.

Participant: Xavier Rival [correspondant].

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.

5.4. ProVerif

Keywords: *formal model*, *security protocols*, *verifier*.

Participants: Bruno Blanchet [correspondant], Xavier Allamigeon [April–July 2004].

PROVERIF (www.proverif.ens.fr) is an automatic security protocol verifier, in the formal model (so called Dolev-Yao model). In this model, cryptographic primitives are considered as black boxes. This protocol verifier is based on an abstract representation of the protocol by Horn clauses. Its main features are:

- It can handle many different cryptographic primitives, including shared- and public-key cryptography (encryption and signatures), hash functions, and Diffie-Hellman key agreements, specified both as rewrite rules or as equations.
- It can handle an unbounded number of sessions of the protocol (even in parallel) and an unbounded message space. This result has been obtained thanks to some well-chosen approximations. This means the verifier can give false attacks, but if it claims that the protocol satisfies some property, then the property is actually satisfied. PROVERIF also provides attack reconstruction: when it cannot prove a property, it tries to reconstruct an attack, that is, an execution trace of the protocol that falsifies the desired property.

The PROVERIF verifier can prove the following properties:

- secrecy (the adversary cannot obtain the secret);
- authentication and more generally correspondence properties, of the form "if an event has been executed, then other events have been executed as well";
- strong secrecy (the adversary does not see the difference when the value of the secret changes);
- equivalences between processes that differ only by terms;

PROVERIF has been used by researchers for studying various kinds of protocols, including electronic voting protocols, certified email protocols, and zero-knowledge protocols. It has been used as a back-end for the tool TULAFALE implemented at Microsoft Research Cambridge, which verifies web services protocols. It has also been used as a back-end for verifying implementations of protocols in F# (a dialect of ML included in .NET), by Microsoft Research Cambridge.

PROVERIF is freely available on the web, at www.proverif.ens.fr, under the GPL license.

5.5. CryptoVerif

Keywords: *computational model*, *security protocols*, *verifier*.

Participant: Bruno Blanchet [correspondant].

CRYPTOVERIF (www.cryptoverif.ens.fr) is an automatic protocol prover sound in the computational model. In this model, messages are bitstrings and the adversary is a polynomial-time probabilistic Turing machine. CRYPTOVERIF can prove:

- secrecy [13];
- correspondences [77], which include in particular authentication; this is the main extension implemented this year.

CRYPTOVERIF provides a generic mechanism for specifying the security assumptions on cryptographic primitives, which can handle in particular symmetric encryption, message authentication codes, public-key encryption, signatures, hash functions.

The generated proofs are proofs by sequences of games, as used by cryptographers. These proofs are valid for a number of sessions polynomial in the security parameter, in the presence of an active adversary. CRYPTOVERIF can also evaluate the probability of success of an attack against the protocol as a function of the probability of breaking each cryptographic primitive and of the number of sessions (exact security).

CRYPTOVERIF is still at a rather early stage of development, but it has already been used for a study of Kerberos in the computational model and a project for using it as a back-end for verifying implementations of protocols in F# is starting at Microsoft Research Cambridge.

CRYPTOVERIF is freely available on the web, at www.cryptoverif.ens.fr, under the CeCILL license.

6. New Results

6.1. Abstract Semantics of Grammars

Keywords: *abstract semantics*, *bottom-up semantics*, *context-free grammar*, *grammar flow analysis*, *grammar problem*, *parsing*, *top-down semantics*.

Participants: Patrick Cousot, Radhia Cousot.

We have introduced abstract interpretations of a fixpoint protoderivation semantics defining the maximal derivations of a transitional semantics of context-free grammars akin to pushdown automata. The result is a hierarchy of bottom-up or top-down semantics refining the classical equational and derivational language semantics and including Knuth grammar problems, classical grammar flow analysis algorithms, and parsing algorithms [84], [15].

6.2. Abstract Semantics of Resolution-Based Logic Languages

Keywords: *Herbrand semantics*, *abstract semantics*, *bottom-up semantics*, *logic programming*, *parsing*, *s-semantics*, *top-down semantics*.

Participants: Patrick Cousot, Radhia Cousot.

The abstract interpretation point of view on context-free grammars has been extended to resolution-based logic programs and proof systems in [58]. Starting from a transition-based small-step operational semantics of PROLOG-like programs (akin to the Warren Machine), we consider maximal infinite derivations for the transition system from most general goals. This semantics is abstracted by instantiation to terms and furthermore to ground terms, following the so called c and s-semantics approach. Orthogonally, these sets of derivations can be abstracted to SLD-trees, call patterns and models, as well as interpreters providing effective implementations (such as PROLOG or lazy PROLOG). These semantics can be presented in bottom-up fixpoint form. This abstract interpretation-based construction leads to classical bottom-up semantics (such as the s-semantics of computed answers of Giorgio Levi, the c-semantics of correct answers of Keith Clark, and the minimal-model semantics of logical consequences of Maarten van Emden and Robert Kowalski). The approach is general and can be applied to infinite and top-down semantics.

6.3. Bi-inductive Definitions and Bifinitary Semantics of the Eager Lambda-Calculus

Keywords: *bi-inductive definition*, *big-step semantics*, *divergence*, *inductive definition*, *natural semantics*, *operational semantics*, *relational semantics*, *small-step semantics*, *structural semantics*.

Participants: Patrick Cousot, Radhia Cousot.

We have introduced an order-theoretic generalization of set-theoretic inductive definitions. This generalization covers inductive, co-inductive, and bi-inductive definitions, including non-monotonic ones, and is preserved by abstraction. This allows the structural operational semantics to describe simultaneously the finite/terminating and infinite/diverging behaviors of programs. This is illustrated on the structural bifinitary semantics of the call-by-value λ-calculus at various levels of abstraction including small/big-step trace/relational/operational semantics [14].

6.4. Verification of Security Protocols in the Formal Model

The formal model of protocols, or Dolev-Yao model is an abstract model in which messages are represented by terms. Our protocol verifier ProVerif relies on this model. The paper by Bruno Blanchet on the verification of correspondence properties in ProVerif [77] written last year is now to appear in the Journal of Computer Security. This year, we have finished our case study of the filesystem Plutus and have implemented several extensions of ProVerif.

6.4.1. Case Study: the Secure Storage System Plutus

Keywords: *automatic verification*, *lazy revocation*, *secure storage*, *security protocols*.

Participants: Bruno Blanchet, Avik Chaudhuri [University of California, Santa Cruz].

We have studied formal security properties of the state-of-the-art protocol for secure file sharing on untrusted storage Plutus, in the automatic protocol verifier ProVerif. As far as we know, this is the first automated formal analysis of a secure storage protocol. The protocol used as the basis of Plutus features a number of interesting schemes like lazy revocation and key rotation. These schemes improve the protocol's performance, but complicate its security properties. Our analysis clarifies several ambiguities in the design and reveals some unknown attacks on the protocol. We propose corrections, and prove precise security guarantees for the corrected protocol [19].

6.4.2. Extensions of ProVerif

Keywords: *attack reconstruction*, *automatic verification*, *security protocols*.

Participant: Bruno Blanchet.

In the frame of a contract with CELAR (see Section 7.6), we have implemented several extensions of ProVerif. We have improved the display of Horn clauses generated by ProVerif and of the derivations from these clauses by providing explanations in English that relate the clauses to the process that represents the protocol. We have implemented the reconstruction of attacks for protocols that rely on weak secrets (such as passwords), so that, when the weak secret is subject to an off-line guessing attack, ProVerif can now, in most cases, provide an explicit description of how to mount the attack in the form of a trace of the process that provides enough knowledge for the adversary to mount the attack. We have also implemented the reconstruction of attacks for strong secrecy. Strong secrecy means that the adversary cannot distinguish when the value of a secret changes. When strong secrecy does not hold, ProVerif can, in most cases, provide a trace up to a point at which the process behaves differently depending on the value of the secret (for instance, a test in the process yields a different result). Note that the existence of such a trace does not necessarily prove that strong secrecy is wrong: the difference of behavior of the process may not be visible to the adversary (for instance, the two branches of a test may not be distinguishable by the adversary). Providing this trace is still very helpful for the user to understand why the proof failed and whether the problem corresponds to a real attack. We have finally improved the display of traces by providing information on where in the process each action of the trace is executed.

6.5. Verification of Security Protocols in the Computational Model

The computational model of protocols considers messages as bitstrings, which is more realistic than the formal model, but also makes the proofs more difficult. Our verifier CryptoVerif is sound in this model. This year, we have finished our case study of Kerberos and have implemented some extensions of CryptoVerif.

6.5.1. Computationally Sound Mechanized Proofs for Basic and Public-key Kerberos

Keywords: *Kerberos*, *automatic verification*, *computational model*, *key usability*, *security protocols*.

Participants: Bruno Blanchet, Aaron Jaggard [Rutgers University], Andre Scedrov [University of Pennsylvania], Joe-Kai Tsay [University of Pennsylvania].

We have done a computationally sound mechanized analysis of Kerberos 5, both with and without its public-key extension PKINIT. We have proved authentication and key secrecy properties using the prover CRYPTOVERIF, which works directly in the computational model; these are the first mechanical proofs of a full industrial protocol at the computational level. We also generalize the notion of key usability and use CRYPTOVERIF to prove that this definition is satisfied by keys in Kerberos [20].

6.5.2. Extensions of CryptoVerif

Keywords: *automatic verification*, *computational model*, *security protocols*.

Participant: Bruno Blanchet.

We have implemented some improvements in the proof strategy of CRYPTOVERIF, in particular to make the proof of public-key protocols fully automatic in most cases. We automate the case distinction between the cases in which the interlocutor is honest or dishonest. We also allow the simplification of games to suggest case distinctions that can lead to further simplifications. Finally, we allow backtracking, since the success or failure of the proof may depend on the order of application of the security assumptions of primitives, when several cryptographic primitives are used.

We also provide a library that defines the most frequent cryptographic primitives, so that the user can include these primitives without having to redefine them.

6.6. Verification of Security Protocols: Formal Model and Computational Model

Keywords: *automatic verification*, *computational model*, *formal model*, *security protocols*.

Participant: Bruno Blanchet.

Bruno Blanchet has written his *habilitation à diriger des recherches* on the verification of security protocols. His report [12] summarizes his work on this topic, which essentially consists in designing and implementing the two protocol verifiers PROVERIF (Section 5.4) and CRYPTOVERIF (Section 5.5). He defended it on November 26, 2008.

6.7. 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 signalling 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).

6.7.1. Reachable Species Analysis

Keywords: *biology*, *protein-protein interaction networks*, *qualitative properties*, *verification*.

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

We have developed an abstract interpretation-based framework to compute an over-approximation of the reachable species in protein-protein interaction networks, that is based on abstractions developed for mobile systems in [89], [90], [7], and [91].

We show several applications of this framework in [92]. In [25], we give a characterization of a class of protein-protein interaction networks for which our analysis is complete. Most networks encountered in signaling fit this characterization, which gives new insights in Systems Biology to robustness and evolvability.

6.7.2. Automatic Reduction of Differential Semantics

Keywords: *biology*, *differential semantics*, *protein-protein interaction networks*, *verification*.

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

We have developed an abstract interpretation-based framework that enables the computation of scalable differential semantics for protein-protein interaction networks. This framework uses indistinguishably techniques in order to detect and prove that some potential correlations between the states of some distinct parts in protein species have no impact on the dynamic of the networks. These information drive the computation of an abstract differential system over a set of self-consistent abstract observables. Results are sound since trajectories in the abstract system are projections of the trajectories in the concrete system. This framework gives new insights in order to describe evolution between systems: indeed several networks can be compared according to the relative amount of control between protein-protein interactions.

This framework has been presented in [60], [62], [64], and [63]. This framework is also submitted to a journal.

6.7.3. Rule-based Modelling of Biological Systems

Keywords: *biology*, *concurrency*, *protein-protein interaction networks*, *refinements*.

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

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 modelling. Refining an interaction allows tuning the kinetics of an interaction according to some constraints in the context of the interacting proteins.

In [24], we propose 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. We use such refinement process in [88] to model a repair scheme for the ADN.

6.8. Relational Inductive Shape Analysis

Keywords: *absence of runtime error*, *inductive definitions*, *memory abstraction*, *relational abstraction*, *shape analysis*, *static analysis*, *symbolic abstract domains*.

Participants: Bor-Yuh Evan Chang [University of California at Berkeley], Xavier Rival.

Developer-supplied data structure specifications are important to shape analyses, as they tell the analysis what information should be tracked in order to obtain the desired shape invariants. We observe that data structure checking code (*e.g.*, used in testing or dynamic analysis) provides shape information that can also be used in static analysis.

We extended the parametric abstract domain proposed in [78] for lightweight automatic shape analysis so as to express properties about relations among shape properties (such as doubly-linked list structures) and also between shape properties and numerical properties (as in search trees or in balanced trees) [21].

The class of inductive definitions that can be used in our analysis is very broad. As a consequence, complex relations among abstract elements sometimes need be infered, so that a powerful reduction operator is needed. We designed a simple such operator [75], which has not been implemented yet. The principle of this operator is to perform an abstract interpretation of inductive definitions using our analysis parametered with other inductive definitions, so as to prove implication properties among them.

6.9. Analysis of Buffer Overruns

Keywords: *automatic verification*, *relational domains*, *static analysis*.

Participants: Pietro Ferrara, Francesco Logozzo [Microsoft Research, Redmond], Manuel Fähndrich [Microsoft Research, Redmond].

We have introduced Stripes, a new relational domain to analyze buffer overruns when directly accessing memory in unsafe code in the .Net framework. We have combined it with both the interval and the linear equalities domain, in order to increase the precision of the analysis. We have implemented it in Clousot, a generic static analyzer; the resulting analysis is both scalable and precise [29].

6.10. Analysis of Multithreaded Programs

We have introduced a generic framework in order to statically analyze multithreaded programs. We have proposed a new property in order to discover unordered communications, and we have applied it to the analysis of Java programs.

6.10.1. Happens-Before Memory Model

Keywords: *automatic verification*, *memory model*, *multithreaded programs*.

Participant: Pietro Ferrara.

Memory models define which behaviors are allowed during the execution of a multithreaded program. We have defined, in a fixpoint form, the happens-before memory model, and then we have abstracted it with a computable semantics. Our approach is generic w.r.t. the programming language, the numerical domain, and the analyzed property [28].

6.10.2. Determinism

Keywords: *automatic verification*, *determinism*, *multithreaded programs*.

Participant: Pietro Ferrara.

Developing and reasoning about multithreaded programs is particularly difficult because of random interleaving during the execution of threads. We have defined a static analysis of a new deterministic property aimed at discovering such behaviors. This property is particularly flexible, and we have proposed some ways of relaxing it [27].

6.10.3. Application to Java Bytecode Language

Keywords: *Java*, *alias analysis*, *automatic verification*, *bytecode*, *multithreaded programs*.

Participant: Pietro Ferrara.

We apply this generic framework to the analysis of Java multithreaded programs at bytecode level. In this context, we have defined an alias analysis in order to precisely trace how threads interact on shared memory and synchronize through monitors [26].

6.11. Boolean Flags in the Polyhedron Abstract Domain

Keywords: *disjunctions*, *linear inequality constraints*, *numerical properties*, *partitioning*, *static analysis*.

Participant: Axel Simon.

In the context of integer linear programming, it is well known that a Boolean variable can be used to express disjunctive information. We generalise this approach to polyhedral analysis by showing that storing a Boolean flag in a bounded $\mathbb{Z}$-polyhedron is equivalent to Boolean partitioning, that is, a Boolean flag can be used to fully distinguish two polyhedra within a single polyhedron [30]. As an example application we show how to model the change of points-to relationships when analyzing loops that operate on string buffers.

6.12. Sound Floating-Point Polyhedra Domain

Keywords: *floating-point arithmetics*, *linear inequality constraints*, *numerical properties*, *static analysis*.

Participants: Liqian Chen, Antoine Miné, Patrick Cousot.

The polyhedron domain is one of the most pervasive numerical abstract domain. Unfortunately, current implementations suffer from the "coefficient explosion" problem causing scalability (when using arbitrary precision rationals) or precision issues (when using machine integers prone to overflow). We have designed [22] a polyhedron domain fully implemented in floating-point arithmetics. The benefits are: a compact representation, efficient algorithms, and a gradual loss of precision. The main challenges were the design of sound and stable algorithms, despite rounding errors. Preliminary experiments show that our domain is on par with state-of-the-art rational implementations (see Section 5.2) when analyzing programs with small integer coefficients, but much more efficient on programs with large or floating-point coefficients.

6.13. A Reduced Disjunctive Extension for Temporal Abstract Domains

Keywords: *abstract domains*, *continuous-time semantics*, *disjunction*, *quasi-synchronous systems*, *static analysis*.

Participant: Julien Bertrane.

The introduction of a syntactic disjunction is crucial in order to have precise abstract operators. It allows indeed a partitioning of sets of behaviors that are too different to be represented by a single abstract element and thus prevents analyses to be blocked. Implementing disjunctions of Abstract Constraints and of Changes Counting elements in the naive way by lifting the domains is however too costly. We introduce a local (on the temporal point of view) disjunction and operators that maintain the fact that constraints never overlap (except in disjunctive branches) so that a path can be found inside disjunctive elements or an abstract operator applied to these elements in a linear time. Reductions between underlying domains (Abstract Constraints and of Changes Counting) are still possible in the extended domain. This extension was introduced in [11].

6.14. Static Analysis of Communicating Imperfectly-Clocked Synchronous Systems Using Continuous-Time Abstract Domains

Keywords: *abstract domains*, *continuous-time semantics*, *quasi-synchronous systems*, *static analysis*.

Participant: Julien Bertrane.

Julien Bertrane has written his PhD on the verification of quasi-synchronous systems. Synchronous programming is a formalism well adapted to the development of computer systems controlling embedded systems. It is based on the idea of the repeated synchrony of some tasks (at clock ticks). However, it is often necessary to use multi-clock systems. Furthermore, physical clocks may desynchronize. Giving such a system a continuous-time semantics does not only enable modeling in a more realistic way the actual execution but also enables defining faster more precise and less costly abstract domains that can prove some of the temporal specifications of such systems. Julien Bertrane's report [11] summarizes his work in this topic. He defended it on November 28, 2008.

7. Contracts and Grants with Industry

7.1. ES_PASS Contract

ES_PASS (Embedded Software Product-based ASSurance) is an ITEA European project grouping technology and tool providers as well as industrial end-users in the field of embedded software for automotive, avionic, railway and space transportation (AbsInt Angewandte Informatik GmbH, Airbus France, CEA/LIST, CS Systèmes d'Information, DaimlerChrysler AG, EADS Astrium SAS, EADS Innovation Works, École Normale Supérieure (ENS), Esterel technologies, FéRIA (IRIT & ONERA), Fraunhofer FIRST, Institut für Bahntechnik (IFB), Saarland University, Siemens VDO, Technical University Munich, Technical University of Madrid, Thales Avionics, Thales Transport). The objective of the participation of the ABSTRACTION project-team to ES_PASS is to confront the ASTRÉE analyzer to a wide range of industrial applications in order to evaluate its practical applicability and prepare its industrialization. Patrick Cousot is the principal investigator for this action.

7.2. SSVAI Contract

SSVAI (Space Software Validation using Abstract Interpretation) is an ESA-ITI project (European Space Agency's Innovative Triangle Initiative) with Astrium Space Transportation, the CEA, the ENS, and the École polytechnique. The activity of the ABSTRACTION project-team in this project is mainly to apply the ASTRÉE static analyzer to the MSU (Monitoring Software Unit) code of the ATV (Automated Transfer Vehicle) for the ISS (International Space Station).

Upon completion of the project, we successfully analyzed several versions of a Scade model of the MSU controller compiled into C (including versions generated by different Scade compilers, and using different generation options). The study demonstrated the ability of ASTRÉE to handle Scade-generated code. It showed that, although the library of abstract domains built in ASTRÉE from our experience on avionics software is sufficient in some cases, achieving zero false alarms would require the development of new abstract domains adapted to the aspects of control theory specific to space control. However, it also showed that this could be mitigated by introducing a few numerical limiters at strategic locations in the code. Patrick Cousot is the principal investigator for this action.

7.3. Asbaprod Contract

ASBAPROD (ASsurance BAsée PRODduit) is an industrial project on static program analysis by abstract interpretation with Airbus France which objective is determined annually.

The main work in 2008 consisted in analyzing with ASTRÉE several new classes of medium-sized (50–100 KLoc) critical embedded C programs: auto-test [70], [71] and communication [72] software. Good results (reasonable analysis time and a couple of remaining false alarms) could be achieved through analysis parametrization. Compared to pure control-command software (the primary target of ASTRÉE), the main challenge is that these are hand-coded (hence exhibiting a wider variety of C constructs and programming patterns) and make use of pointers and pointer arithmetics (although there is no dynamic memory allocation). This diversification helped strengthen ASTRÉE (in particular its new memory model [93]) and enlarge its focus. Patrick Cousot is the principal investigator for this action.

7.4. Controvert ANR

The CONTROVERT project (2005–2008) brings together control-theory researchers of ONERA/DCSD and the Université Paul Sabatier of Toulouse and computer scientists from the ABSTRACTION project-team. A first objective is to bridge the gap between control-theory-based methods for analyzing properties of models of systems and their controllers (e.g. robustness) by continuous Lagrangian overapproximation of the system trajectories and abstract-interpretation-based methods for analyzing control/command programs (e.g. safety properties) in opened loop. A second objective is to use the results of the control-command theoretic analysis of the closed loop to support the program analysis in the context of the controlled system. Patrick Cousot is the principal investigator for this action.

7.5. FormaCrypt ARA

The ABSTRACTION project-team coordinates the FORMACRYPT project, on "formal proofs and probabilistic semantics in cryptography" (project web site: http://www.di.ens.fr/~blanchet/formacrypt/index.html). This project is financed by the *Agence Nationale pour la Recherche*, in the frame of the *Action de Recherche Amont Sécurité, Systèmes embarqués et Intelligence Ambiante (ARA SSIA)*. This project of a duration of 3 years and half (January 2006–July 2009) brings together researchers of the INRIA project-teams ABSTRACTION and CASCADE (LIENS, *Laboratoire d'Informatique de l'École Normale Supérieure*), SECSI (LSV, *Laboratoire Spécification et Vérification, ENS Cachan*), and CASSIS (LORIA, *Laboratoire Lorrain de Recherche en Informatique et ses Applications*), as well as Martín Abadi as scientific advisor. The goal of this project is to bridge the gap between the formal and computational models of security protocols, so as to obtain automatic proofs of protocols valid in the computational model. This project has led to 30 publications in international

conferences, workshops, and journals, to the implementation of two tools, CRYPTOVERIF and an extension of AVISPA, and strongly contributed to the organisation of a series of international workshops (the workshops on Formal and Computational Cryptography, FCC). Bruno Blanchet is the principal investigator for this action.

7.6. Contract with CELAR

In the frame of a 2-year contract (July 2008–October 2010) with CELAR (*Centre d'Electronique de l'Armement*), we implement extensions of PROVERIF in order to make it easier to use: extensions to attack reconstruction, improvements to the user interface and to the documentation. Bruno Blanchet is the principal investigator for this action.

7.7. Thésée ANR

The objective of the THÉSÉE project (2006–2009) is to develop static analysis techniques for proving the absence of runtime errors in asynchronous (real-time) programs. The project is in cooperation with EDF and Airbus France. The main problem is to scale up traditional sequential static analysis methods so as to cope with the combinatorial explosion resulting form the interleaving of communications and interactions through shared variables in a parallel execution of the asynchronous processes. Patrick Cousot is the principal investigator for this action.

8. Dissemination

8.1. Interaction with the Scientific Community

8.1.1. 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),

8.1.2. Collective Responsibilities

Bruno Blanchet is a member of the *commission de spécialistes* (hiring committee) of ENS Cachan.

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

Radhia Cousot is head of the Abstract Interpretation group at École Polytechnique under a convention between the CNRS, the École Normale Supérieure and the École Polytechnique.

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

8.1.3. Editorial Boards and Program Committees

Bruno Blanchet is associate editor of the International Journal of Applied Cryptography (IJACT).

Bruno Blanchet was member of the program committee of the ACM SIGPLAN - SIGACT Symposium on Principles of Programming Languages (POPL'09), the ACM SIGPLAN Workshop on Programming Languages and Analysis for Security (PLAS 2008), the IEEE Computer Security Foundations Symposium (CSF 2008), and program committee co-chair of the 4th Workshop on Formal and Computational Cryptography (FCC 2008).

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 First International Conference on Foundations of Informatics, Computing, and Software (FICS 2008), the sixth IEEE International Conferences on Software Engineering and Formal Methods (SEFM-08), and the second IFIP Working Conference on Verified Software: Theories, Tools, and Experiments (VSTTE 2008)

Radhia Cousot is member of the advisory board of the Higher-Order Symbolic Computation journal (HOSC, Springer).

Radhia Cousot was member of the program committee of the Ninth Verification, Model Checking and Abstract Interpretation (VMCAI'08) international conference and the Fifteen International Static Analysis Symposium (SAS 2008).

Jérôme Feret participated in the expert committee of the sixth call for projects of the National Foundation of Research in Aeronautic and Space Applications (FNRAE).

Antoine Miné was member of the program committee of the 16th International Conference on Compiler Construction (CC 2008).

Xavier Rival was member of the program committee of the ACM Conference on Languages, Compilers, and Tools for Embedded Systems (LCTES 2008) and the Bytecode 2008 Workshop.

8.1.4. PhD and Habilitation Juries

Patrick Cousot was the director of the PhD thesis of Julien Bertrane (École polytechnique, November 2008).

Patrick Cousot was the director of the HdR of Bruno Blanchet (University Paris-Dauphine, November 2008).

Radhia Cousot was reviewer of the PhD thesis of Greta Yorsh (Tel Aviv University, March 2008).

8.2. Teaching

8.2.1. Supervision of PhDs and Internships

Patrick Cousot supervised the PhD thesis of Julien Bertrane and the research apprenticeships of Ferdinanda Camporesi and David Durrleman. Patrick Cousot and Antoine Miné supervised the research apprenticeship of Liqian Chen. Radhia Cousot supervised the PhD thesis of Guillaume Capron and the PhD thesis of Pietro Ferrara.

8.2.2. Training

Participants: Patrick Cousot, Radhia Cousot, Laurent Mauborgne, Antoine Miné, Xavier Rival.

The ABSTRACTION project-team organised a one-day training session on the ASTRÉE static analyzer (see Section 5.1) for academic and industrial partners in the ES_PASS project (see Section 7.1) and well as to a two days training in abstract interpretation for the Chinese Delegation of the East China Normal University, Shanghai, China in « Highly Dependable Systems »[42].

8.2.3. Graduate Courses

Bruno Blanchet taught 6 hours in the MPRI (Master Parisien de Recherche en Informatique) course on Cryptographic protocols: formal and computational proofs of Stéphanie Delaune and Cédric Fournet [36].

Radhia Cousot was responsible of the M2 course "Abstract interpretation: application to verification and static analysis" at the MPRI (Master Parisien de Recherche en Informatique) [57]. Julien Bertrane, Patrick Cousot, Jérôme Feret, Laurent Mauborgne, Antoine Miné, and Xavier Rival participated in the course.

Pietro Ferrara taught 16 hours in course on Analysis and Verification of Programs (prof. Agostino Cortesi) of the Master program in Computer Science of University Ca' Foscari of Venice.

8.2.4. Undergraduate Courses

Julien Bertrane was teaching assistant at École Polytechnique.

Patrick Cousot gave the M1 course "Foundations of abstract interpretation: application to semantics" [46] at the École Normale Supérieure.

Laurent Mauborgne was a partial time associate professor (*professeur chargé de cours*) at École Polytechnique. He gave a 32-hour course on static analysis for 3rd year students (M1) [68]. He gave 30 hours of lectures in small groups for 2nd year students, following the course "Foundations of Computer Science" directed by François Morain and Jean-Marc Steyaert.

Xavier Rival gave training sessions on "Algorithmics and programming in Java" and 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.

8.3. PhD theses

Julien Bertrane defended his *PhD* on November 28, 2008 on static analysis of communicating imperfectly-clocked synchronous systems using continuous-time abstract domains [11].

8.4. Habilitation theses

Bruno Blanchet defended his *habilitation à diriger des recherches* on November 26, 2008 on the verification of security protocols [12].

8.5. Monographs

8.5.1. Value-Range Analysis of C Programs

Research into new analysis techniques is often hampered by the technical difficulties of analyzing accesses through pointers, pointer arithmetic, coercion between types, integer wrap-around and other low-level behaviour. While approaches to some of these problems can be found in the available literature, there is no coherent treatment that addresses all of these aspects together.

This book presents a concise, yet formal description of a value-range analysis that soundly approximates the semantics of C programs using systems of linear inequalities (polyhedra). The analysis is formally specified down to the bit-level while providing a precise approximation of all low-level aspects of C using polyhedral operations. As such, it provides a basis for implementing new analyses that are aimed at verifying higher-level program properties precisely. One such analysis, namely the tracking of the NUL position in C string buffers, is presented, thereby demonstrating the extensibility of the approach.

While the book focuses on a sound analysis of C, it will be useful to any researcher and student with an interest in static analysis of real-world programming languages [31]. It should be particularly valuable to Ph.D. students embarking on a related topic.

8.6. Participation in Conferences and Seminars

8.6.1. Participation in Conferences

VMCAI: International Conference on Verification, Model Checking and Abstract Interpretation (San Francisco, January 2008).
Julien Bertrane, Bruno Blanchet, Patrick Cousot, Radhia Cousot, Jérôme Feret, Laurent Mauborgne, Antoine Miné, and Xavier Rival attended the conference. Patrick Cousot chaired a session. Radhia Cousot gave an invited talk [23]. Jérôme Feret presented [25].

30YAI: 30 Years of Abstract Interpretation Workshop (San Francisco, January 2008).
Julien Bertrane, Bruno Blanchet, Patrick Cousot, Radhia Cousot, Jérôme Feret, Laurent Mauborgne, Antoine Miné, and Xavier Rival attended the workshop. Patrick Cousot gave an invited talk [53].

POPL: ACM Symposium on Principles of Programming Languages (San Francisco, January 2008).
Julien Bertrane, Bruno Blanchet, Patrick Cousot, Radhia Cousot, Jérôme Feret, Laurent Mauborgne, Antoine Miné, and Xavier Rival attended the conference.

AsiaCCS: ACM Symposium on Information, Computer and Communications Security (Tokyo, Japan, March 2008).
Bruno Blanchet attended the conference.

ESOP: European Symposium on Programming (Budapest, Hungary, March 2008).
Pietro Ferrara attended the conference.

TACAS: International Conference on Tools and Algorithms for the Construction and Analysis of Systems (Budapest, Hungary, March 2008).
Pietro Ferrara attended the conference.

FASE: International Conference on Fundamental Approaches to Software Engineering (Budapest, Hungary, March 2008).
Pietro Ferrara attended the conference.

FOSSACS: International Conference on Foundations of Software Science and Computation Structures (Budapest, Hungary, April 2008).
Pietro Ferrara attended the conference.

CC: International Conference on Compiler Construction (Budapest, Hungary, April 2008).
Pietro Ferrara attended the conference. Antoine Miné attended the conference and chaired a session.

Bytecode: Workshop on Bytecode Semantics, Verification, Analysis and Transformation (Budapest, Hungary, April 2008).
Pietro Ferrara presented [26].

TAP: International Conference on Tests and Proofs (Prato, Italy, April 2008).
Pietro Ferrara presented [28].

PLDI: ACM Conference on Programming Language Design and Implementation (PLDI) (Tucson, USA, June 2008).
Xavier Rival attended the conference.

LCTES: ACM Conference on Languages, Compilers, and Tools for Embedded Systems (Tucson, USA, June 2008).
Xavier Rival attended the conference and chaired a session.

FCS-ARSPA-WITS: Joint Workshop on Foundations of Computer Security, Automated Reasoning for Security Protocol Analysis and Issues in the Theory of Security (Pittsburgh, PA, USA, June 2008).
Bruno Blanchet attended the workshop.

FCC: Workshop on Formal and Computational Cryptography (Pittsburgh, PA, USA, June 2008).
Bruno Blanchet chaired a session.

FICS: First International Conference on Foundations of Informatics, Computing and Software (Shanghai, China, 03–06 June 2008).
Patrick Cousot gave an invited talk [45] and chaired a session. Julien Bertrane attended the conference.

CSF: Computer Security Foundations Symposium (Pittsburgh, PA, USA, June 2008).
Bruno Blanchet chaired a session and gave a 5 minutes talk on [20].

NSV: First International Workshop on Numerical Abstractions for Software Verification (Princeton, NJ, USA, July 2008).
Patrick Cousot gave an invited talk [56] and chaired a session.

IFIP WG 2.3 Cambridge meeting: (Cambridge University, UK, July 2008).
Patrick Cousot attended the semestrial meeting and gave a presentation [55].

OOPSLA: ACM Conference on Object-oriented Programming (Nashville, USA, October 2008).
Pietro Ferrara presented [29].

VSTTE'08: Second IFIP Working Conference on Verified Software: Theories, Tools, and Experiments (Toronto, Canada, October 2008).
Xavier Rival attended the conference.

SEFM: IEEE International Conference on Software Engineering and Formal Methods (Cape Town, South Africa, November 2008).
Pietro Ferrara presented [27].

APLAS: Asian Symposium on Programming Languages and Systems (Bangalore, India, December 2008).
Antoine Miné attended the conference.

8.6.2. Invitations and Participation in Seminars

Julien Bertrane gave seminars and presentations on static analysis of imperfectly-clocked synchronous systems in the Logic and Semantics Group of the Queen Mary University (London) [33], in the Dipartimento di Informatica of the Università degli Studi di Verona [34] and in the Compiler Design Lab of the Universität des Saarlandes (Saarbrücken) [35].

Bruno Blanchet presented a talk on "Automated Security Proofs with Sequences of Games" (joint work with David Pointcheval) at IRIT, Toulouse, France, January 2008 [38] and at RCIS, AIST, Tokyo, Japan, March 2008 [37].

Patrick Cousot gave seminars and presentations at Stony Brook University [51], the Center for Computational and Systems Biology, The Microsoft Research — University of Trento, Trento, Italy [50], the Dipartimento di Mathematica Pura ed Applicata, Universitá degli Studi di Padova, Italy [44], the Collège de France [47], Schloß Dagstuhl, Wadern, Germany [54], New York University [48], [43], the Comité de Direction de l'INRIA [49], the IFIP WG 2.3 working group [55], the Seoul National University, Korea [40], the Max Planck Institut für Informatik, Saarbrücken [52], the ESA/ Noordwijk (ESTEC), the Netherland [41], the Airbus workshop on formal verification tools strategy [39].

Jérôme Feret presented an abstract interpretation framework for analyzing qualitative properties of biological pathways [59] and a framework to represent biochemical processes in the π-calculus [61] in working groups at Harvard Medical School. He presented an abstract interpretation framework for reducing differential semantics for biological networks in a working group at Harvard Medical School [60] and in seminars at Verimag [62], at Paris VII [64], and at the École Normale Supérieure [63].

Pietro Ferrara presented a talk on static analysis via abstract interpretation of the happens-before memory model at the University "Ca' Foscari" of Venice [66] and at the École Normale Supérieure [67]. He presented a talk on static analysis of the determinism of multithreaded programs at the University "Ca' Foscari" of Venice [65].

Antoine Miné participated in the seminar "Beyond the Finite: New Challenges in Verification and Semistructured Data", Schloss Dagstuhl, Wadern, Germany, April 2008. Antoine Miné presented some results of the SSVAI project (see Section 7.2) at ESTEC [73], Noordwijk, Netherlands, September 2008. Antoine Miné participated in the workshop "Security and Reliability of Software Systems" in Bangalore, India, December 2008, and presented a 2-hour invited tutorial [69] on ASTRÉE.

Xavier Rival attended the "THEORY Workshop at VSTTE 2008" and gave a one hour invited lecture on parametric abstract domains for shape analysis [75]. Xavier Rival participated in the seminar "Emerging Uses and Paradigms for Dynamic Binary Translation", Shhloss Dagstuhl, Wadern, Germany, October 2008, and gave a lecture on certified compilation [74].

Axel Simon participated in the seminar “Beyond the Finite: New Challenges in Verification and Semistructured Data”, Schloss Dagstuhl, Wadern, Germany, April 2008. He presented work on “Scalable Program Analysis using Convex Polyhedra” to the Theoretical Computer Science group at the University of Kent, November 2008 [76].

8.7. Short-Term Visitors

Agostino Cortesi (Università di Venezia) visited the project-team Abstraction in April 2008.

Reinhard Wilhelm (Universitët des Saarlandes) and Francesco Ranzato (Università di Podova) came in June 2008.

Neil Jones (University of Copenhagen), Jifeng He (East China Normal University), Ji Wang (National University of Defense Technology), Jing Liu (East China Normal University), Huibiao Zhu (East China Normal University), Dehui Du (East China Normal University), Jian Guo (East China Normal University), and Wei Dong (National University of Defense Technology) came in September 2008.

9. Bibliography

Major publications by the team in recent years

[1] B. BLANCHET. *A Computationally Sound Mechanized Prover for Security Protocols*, in "IEEE Transactions on Dependable and Secure Computing", vol. 5, n^{o} 4, October–December 2008, p. 193–207.

[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), San Diego, California, USA", 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", vol. 277, n^{o} 1–2, 2002, 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, Boston, Massachusetts, United States", ACM Press, New York, 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, Portland, Oregon, United States", ACM Press, New York, 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, Los Angeles, California", ACM Press, New York, NY, 1977, p. 238–252.

[7] J. FERET. *Abstract Interpretation of Mobile Systems*, in "Journal of Logic and Algebraic Programming, special issue on pi-Calculus", vol. 39, n^{o} 1, 2005.

[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, vol. 3444, Springer-Verlag, 2005, p. 5–20.

[9] A. MINÉ. *The Octagon Abstract Domain*, in "Higher-Order and Symbolic Computation", vol. 19, 2006, 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, Venice, Italy", ACM Press, New York, New York, United States, 2004, p. 1–13.

Year Publications

Doctoral Dissertations and Habilitation Theses

[11] J. BERTRANE. *Static analysis of communicating imperfectly-clocked synchronous systems using continuous-time abstract domains*, Thèse, École polytechnique, 28 Novembre 2008.

[12] B. BLANCHET. *Vérification automatique de protocoles cryptographiques : modèle formel et modèle calculatoire*, Habilitation à Diriger des Recherches, Université Paris-Dauphine, November 2008.

Articles in International Peer-Reviewed Journal

[13] B. BLANCHET. *A Computationally Sound Mechanized Prover for Security Protocols*, in "IEEE Transactions on Dependable and Secure Computing", vol. 5, n^{o} 4, October–December 2008, p. 193–207.

[14] P. COUSOT, R. COUSOT. *Bi-inductive Structural Semantics*, in "Information and Computation", To appear, 2008.

[15] P. COUSOT, R. COUSOT. *Grammar Semantics, Analysis, and Parsing by Abstract Interpretation*, in "Theoretical Computer Science", To appear, 2008.

[16] P. COUSOT, R. COUSOT, J. FERET, L. MAUBORGNE, A. MINÉ, X. RIVAL. *Why does* ASTRÉE *scale up?*, in "Formal Methods in Systems Design", 2008.

[17] M. HINCHEY, M. JACKSON, P. COUSOT, B. COOK, J. P. BOWEN, T. MARGARIA. *Software engineering and formal methods*, in "Communications of the ACM", vol. 51, n^{o} 9, September 2008, p. 54–59.

[18] D. MONNIAUX. *The pitfalls of verifying floating-point computations*, in "ACM Transactions on Programming Languages and Systems", vol. 30, 3, 2008.

International Peer-Reviewed Conference/Proceedings

[19] B. BLANCHET, A. CHAUDHURI. *Automated Formal Analysis of a Protocol for Secure File Sharing on Untrusted Storage*, in "IEEE Symposium on Security and Privacy, Oakland, CA", IEEE, May 2008, p. 417–431.

[20] B. BLANCHET, A. D. JAGGARD, A. SCEDROV, J.-K. TSAY. *Computationally Sound Mechanized Proofs for Basic and Public-key Kerberos*, in "ACM Symposium on Information, Computer and Communications Security (ASIACCS'08), Tokyo, Japan", ACM, March 2008, p. 87–99.

[21] B.-Y. E. CHANG, X. RIVAL. *Relational Inductive Shape Analysis*, in "ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, San Francisco, USA", ACM, January 2008, p. 247–260.

[22] 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), Bangalore, India", Lecture Notes in Computer Science, vol. 5356, Springer, December 2008, p. 3–18.

[23] R. COUSOT. *Abstract Interpretation of Non-monotone Bi-inductive Semantic Definitions*, in "Proceedings of the Ninth International Conference on Verification, Model Checking and Abstract Interpretation, VMCAI'2008, San Francisco, USA", F. LOGOZZO, D. A. PELED, L. D. ZUCK (editors), Lecture Notes in Computer Science, vol. 4905, Springer, Berlin, Germany, 7–9 January 2008, p. 1–3.

[24] 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, vol. 5054, Springer, Berlin, Germany, 4–5 June 2008, p. 103–122.

[25] V. DANOS, J. FERET, W. FONTANA, J. KRIVINE. *Abstract interpretation of cellular signalling networks*, in "Proceedings of the Ninth International Conference on Verification, Model Checking and Abstract Interpretation, VMCAI'2008, San Francisco, USA", F. LOGOZZO, D. A. PELED, L. D. ZUCK (editors), Lecture Notes in Computer Science, vol. 4905, Springer, Berlin, Germany, 7–9 January 2008, p. 83–97.

[26] P. FERRARA. *A fast and precise alias analysis for data race detection*, in "Proceedings of the Third Workshop on Bytecode Semantics, Verification, Analysis and Transformation (Bytecode'08)", Electronic Notes in Theoretical Computer Science, Elsevier, April 2008.

[27] P. FERRARA. *Static analysis of the determinism of multithreaded programs*, in "Proceedings of the Sixth IEEE International Conference on Software Engineering and Formal Methods (SEFM 2008)", A. CERONE, S. GRUNER (editors), IEEE Computer Society, November 2008.

[28] P. FERRARA. *Static analysis via abstract interpretation of the happens-before memory model*, in "Proceedings of the Second International Conference on Tests and Proofs (TAP'08)", B. MEYER, R. HÄHNLE (editors), Lecture Notes in Computer Science, vol. 4966, Springer, April 2008, p. 116–133.

[29] P. FERRARA, F. LOGOZZO, M. FÄHNDRICH. *Safer Unsafe Code for .NET*, in "OOPSLA '08: Proceedings of the 23rd ACM SIGPLAN conference on Object oriented programming systems languages and applications, New York, NY, USA", ACM, 2008, p. 329–346.

[30] A. SIMON. *Splitting the Control Flow with Boolean Flags*, in "Static Analysis Symposium, Valencia, Spain", M. ALPUENTE, G. VIDAL (editors), LNCS, vol. 5079, Springer, July 2008, p. 315-331.

Scientific Books (or Scientific Book chapters)

[31] A. SIMON. *Value-Range Analysis of C Programs*, Springer, August 2008.

Research Reports

[32] P. COUSOT, R. COUSOT, J. FERET, L. MAUBORGNE, A. MINÉ, X. RIVAL, E.-J. SIMS. *Mode d'emploi et manuel de référence de l'analyseur* ASTRÉE, 319 pages, 2008.

Other Publications

[33] J. BERTRANE. *Reduced product of abstract domains for the static analysis of properties of imperfectly-clocked synchronous systems*, Seminar: Logic and Semantics Group of the Queen Mary University (London), 5 August 2008.

[34] J. BERTRANE. *Static analysis by abstract interpretation of properties of imperfectly-clocked synchronous systems*, Seminar: Dipartimento di Informatica of the Università degli Studi di Verona, 16 September 2008.

[35] J. BERTRANE. *Static analysis of communicating imperfectly-clocked synchronous systems using continuous-time abstract domains*, Seminar: Compiler Design Lab of the Universität des Saarlandes (Saarbrücken), 18 November 2008.

[36] B. BLANCHET, S. DELAUNE, C. FOURNET. *Cryptographic protocols: formal and computational proofs*, M2 course of the MPRI (Master Parisien de Recherche en Informatique), 2008.

[37] B. BLANCHET, D. POINTCHEVAL. *Automated Security Proofs with Sequences of Games*, Seminar, RCIS, AIST, Tokyo, Japan, March 2008.

[38] B. BLANCHET, D. POINTCHEVAL. *Automated Security Proofs with Sequences of Games*, Seminar, IRIT, Toulouse, France, January 2008.

[39] P. COUSOT. AS-TRÉE *and related works*, Airbus Workshop on Formal Verification Tools Strategy, Airbus France, Toulouse, 4–5 December 2008.

[40] P. COUSOT. *Abstract Interpretation and Application to Static Analysis of Safety-Critical Embedded Computer Software*, Computer Science & Engineering Distinguished Lecture Series, Seoul National University, Seoul, Korea, 30 September 2008.

[41] P. COUSOT. *Abstract Interpretation and Application to the Static Analysis of Mission-Critical Embedded Computer Software*, Final review of the ESA ITI project « Space Software Validation using Abstract Intepretation », ESA/ Noordwijk (ESTEC), The Netherlands, 19 September 2008.

[42] P. COUSOT. *Abstract Interpretation and its Applications*, Series of seminars for the Chinese Delegation of ECNU, Shanghai, China in « Highly Dependable Systems », ENS, Paris, 17–18 September 2008.

[43] P. COUSOT. *Abstract-Interpretation-based Static Analysis of Safety-Critical Embedded Software*, Computer Science Colloquium, Courant Institute of Mathematical Sciences, New York University, 21 November 2008.

[44] P. COUSOT. *Advances and Challenges in Static Program Analysis by Abstract Interpretation*, Colloquia Patavina, Dipartimento di Mathematica Pura ed Applicata, Universitá degli Studi di Padova, Italy, 19 February 2008.

[45] P. COUSOT. *Automatic Software Verification by Abstract Interpretation*, First International Conference on Foundations of Informatics, Computing and Software, Shanghai, China, 03–06 June 2008.

[46] P. COUSOT. *Foundations of abstract interpretation: application to semantics*, M1 course of the École Normale Supérieure, 2008.

[47] P. COUSOT. *La vérification des programmes par interprétation abstraite*, Séminaire de la Chaire d'innovation technologique Liliane Bettencourt, Collège de France, Paris, 22 February 2008.

[48] P. COUSOT. *Parameterized Refinement in Abstract-Interpretation-Based Static Analysis*, Verification Seminar, Courant Institute of Mathematical Sciences, New York University, 30 April 2008.

[49] P. COUSOT. *Présentation de l'équipe/projet «* ABSTRACTION *» de l'INRIA Paris-Rocquencourt commune au CNRS et à l'ENS*, Comité de Direction de l'INRIA, Paris, 10 June 2008.

[50] P. COUSOT. *Software Verification by Abstract Interpretation and the* ASTRÉE *Static Analyzer*, Center for Computational and Systems Biology, The Microsoft Research — University of Trento, Trento, Italy, 5 February 2008.

[51] P. COUSOT. *Software Verification by Abstract Interpretation and the* ASTRÉE *Static Analyzer*, Computer Science Department, Stony Brook University, USA, 18 January 2008 2008.

[52] P. COUSOT. *Static Software Analysis, in the Large*, Distinguished Lecture Series at the MPI for Software Systems, Max Planck Institut für Informatik, Saarbrücken, 26 August 2008.

[53] P. COUSOT. *Thirty Years of Abstract Interpretation*, POPL'08 Workshop on « Thirty Years of Abstract Interpretation », David Schmidt and Roberto Giacobazzi, San Francisco, USA, 9 January 2008.

[54] P. COUSOT. *Why does* ASTRÉE *scale up?*, Dagstuhl seminar 08161, Scalable Program Analysis, Schloß Dagstuhl, Wadern, Germany, 15 April 2008.

[55] P. COUSOT, R. COUSOT. *Calculational Design of Semantics of the Eager Lambda-Calculus by Abstract Interpretation*, IFIP WG 2.3 meeting, Cambridge University, UK, 25 July 2008.

[56] P. COUSOT, R. COUSOT. *Numerical Domains for Software Verification by Abstract Interpretation*, First International Workshop on Numerical Abstractions for Software Verification, Princeton, New Jersey, USA, 8 July 2008.

[57] P. COUSOT, R. COUSOT, J. FERET, L. MAUBORGNE, A. MINÉ, X. RIVAL. *Foundations of abstract interpretation: application to semantics*, M2 course of the MPRI (Master Parisien de Recherche en Informatique), 2008.

[58] P. COUSOT, R. COUSOT, R. GIACOBAZZI. *Abstract Interpretation of Resolution-Based Semantics*, To appear, 2008.

[59] J. FERET. *Abstract Interpretation of Biological Networks*, Working group: Fontana Lab, Harvard Medial School, Boston, USA, 17 January 2008.

[60] J. FERET. *Abstract Interpretation of Differnetial Semantics for Biological Networks*, Working group: Fontana Lab, Harvard Medial School, Boston, USA, 22 May 2008.

[61] J. FERET. *Representation of biochemical processes using the π-calculus*, Lecture group: Fontana Lab, Harvard Medial School, Boston, USA, 13 Mars 2008.

[62] J. FERET. *Réduction de sémantiques différentielles pour réseaux d'interactions entre protéines par interprétation abstraite*, Séminaire Verimag, 13 Novembre 2008.

[63] J. FERET. *Réduction de sémantiques différentielles pour réseaux d'interactions entre protéines par interprétation abstraite*, Séminaire de l'Équipe Sémantique et Interprétation Abstraite, 12 décembre 2008.

[64] J. FERET. *Réduction de sémantiques différentielles pour réseaux d'interactions entre protéines par interprétation abstraite*, Séminaire de l'Équipe PPS, 20 Novembre 2008.

[65] P. FERRARA. *Static analysis of the determinism of multithreaded programs*, Lunch Seminar: University "Ca' Foscari" of Venice, Italy, 18 December 2008.

[66] P. FERRARA. *Static analysis via abstract interpretation of the happens-before memory model*, Lunch Seminar: University "Ca' Foscari" of Venice, Italy, 28 May 2008.

[67] P. FERRARA. *Static analysis via abstract interpretation of the happens-before memory model*, Séminaire de l'Équipe Sémantique et Interprétation Abstraite, 20 juin 2008.

[68] L. MAUBORGNE. *Static Analysis of Programs*, M1 course of the École Polytechnique, 2008.

[69] A. MINÉ. *Building a specialized static analyzer: the* ASTRÉE *experience*, Workshop on Security and Reliability of Software Systems, Bangalore, India, 12 December 2008.

[70] A. MINÉ. *Rapport technique sur l'analyse de l'autotest A340 par* ASTRÉE, Deliverable for the Asbaprob contract, 15 March 2008.

[71] A. MINÉ. *Rapport technique sur l'analyse de l'autotest A380 par* ASTRÉE, Deliverable for the Asbaprob contract, 15 March 2008.

[72] A. MINÉ. *Rapport technique sur l'analyse de la passerelle par* ASTRÉE, Deliverable for the Asbaprob contract, 15 March 2008.

[73] A. MINÉ. *The* ASTRÉE *Static Analyzer, Space Software Validation using Abstract Interpretation*, Presentation at ESTEC, Noordwijk, Netherlands, 19 September 2008.

[74] X. RIVAL. *Certified Compilation*, Presentation at the Dagstuhl Seminar 08441, Emerging Uses and Paradigms for Dynamic Binary Translation, 30 October 2008.

[75] X. RIVAL. *Design of parametric abstract domains for shape analysis*, Presentation at The THEORY Workshop at VSTTE 2008, 9 October 2008.

[76] A. SIMON. *Scalable Program Analysis using Convex Polyhedra*, Theoretical Computer Science group at the University of Kent, UK, November 2008.

References in notes

[77] B. BLANCHET. *Computationally Sound Mechanized Proofs of Correspondence Assertions*, in "20th IEEE Computer Security Foundations Symposium (CSF'07), Venice, Italy", IEEE, July 2007, p. 97–111.

[78] B.-Y. E. CHANG, X. RIVAL, G. C. NECULA. *Shape Analysis with Structural Invariant Checkers*, in "Proceedings of the Fourteenth International Symposium on Static Analysis, SAS'07, Kongens Lyngby, Denmark", G. FILÉ, H. RIIS-NIELSON (editors), Lecture Notes in Computer Science, vol. 4634, Springer, Berlin, Germany, 22–24 August 2007, p. 384-401.

[79] 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, NY, USA, 2007, p. 7–9.

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

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

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

[83] P. COUSOT, R. COUSOT. *Bi-inductive Structural Semantics*, in "Structural Operational Semantics, SOS'07, Wroclaw, Poland", R. J. VAN GLABBEEK, M. HENNESSY (editors), ENTCS (1)., vol. 192, n^{o} 1, Elsevier B.V., 9 July 2007.

[84] 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, vol. 4444, Springer, Berlin, Germany, 2007.

[85] 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, San Antonio, Texas", ACM Press, New York, New York, United States, 1979, p. 269–282.

[86] 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, vol. 3444, Springer, Berlin, Germany, 2–10 April 2005, p. 21–30.

[87] 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, Shanghai, China", M. HINCHEY, J. HE, J. SANDERS (editors), IEEE Computer Society Press, Los Alamitos, California, USA, 6–8 June 2007.

[88] V. DANOS, J. FERET, W. FONTANA, J. KRIVINE, R. HARMER. *Investigation of a biological repair scheme*, in "Proceedings of the ninth Workshop on Membrane Computing, WMC9, Edinburgh, UK, Edinburgh, UK", G. PAUN (editor), LNCS, to appear, n^{o} 5391, Springer 2009, Berlin, Germany, 28–31 July 2008.

[89] J. FERET. *Confidentiality Analysis of Mobile Systems*, in "Seventh International Static Analysis Symposium (SAS'00)", LNCS, Springer-Verlag, n^{o} 1824, Springer-Verlag, 2000.

[90] J. FERET. *Dependency analysis of Mobile Systems*, in "European Symposium on Programming (ESOP'02)", LNCS, Springer-Verlag, n^{o} 2305, Springer-Verlag, 2002.

[91] J. FERET. *Analysis of Mobile Systems by Abstract Interpretation*, Thèse de doctorat en informatique, École Polytechnique, 25 February 2005.

[92] J. FERET. *Reachability Analysis of Biological Signalling Pathways by Abstract Interpretation*, in "Proceedings of the International Conference of Computational Methods in Sciences and Engineering, ICCMSE'2007, Corfu, Greece, Corfu, Greece", T. E. SIMOS (editor), American Institute of Physics Conference Proceedings, vol. 2, n^{o} 963, American Institute of Physics, 25–30 September 2007, p. 619–622.

[93] A. MINÉ. *Field-Sensitive Value Analysis of Embedded C Programs with Union Types and Pointer Arithmetics*, in "Proceedings of the ACM SIGPLAN/SIGBED Conference on Languages, Compilers, and Tools for Embedded Systems, LCTES'2006", ACM Press, New York, USA, June 2006, p. 54–63.