RESEARCH CENTER

FIELD
Algorithmics, Programming, Software and Architecture

Activity Report 2017

Section Software

Edition: 2018-02-19

ARIC Project-Team

6. New Software and Platforms

6.1. FPLLL

KEYWORDS: Euclidean Lattices - Computer algebra system (CAS) - Cryptography
SCIENTIFIC DESCRIPTION: The fplll library is used or has been adapted to be integrated within several mathematical computation systems such as Magma, Sage, and PariGP. It is also used for cryptanalytic purposes, to test the resistance of cryptographic primitives.
FUNCTIONAL DESCRIPTION: fplll contains implementations of several lattice algorithms. The implementation relies on floating-point orthogonalization, and LLL is central to the code, hence the name.

It includes implementations of floating-point LLL reduction algorithms, offering different speed/guarantees ratios. It contains a 'wrapper' choosing the estimated best sequence of variants in order to provide a guaranteed output as fast as possible. In the case of the wrapper, the succession of variants is oblivious to the user.

It includes an implementation of the BKZ reduction algorithm, including the BKZ-2.0 improvements (extreme enumeration pruning, pre-processing of blocks, early termination). Additionally, Slide reduction and self dual BKZ are supported.

It also includes a floating-point implementation of the Kannan-Fincke-Pohst algorithm that finds a shortest non-zero lattice vector. For the same task, the GaussSieve algorithm is also available in fplll. Finally, it contains a variant of the enumeration algorithm that computes a lattice vector closest to a given vector belonging to the real span of the lattice.

- Author: Damien Stehlé
- Contact: Damien Stehlé
- URL: https://github.com/fplll/fplll

6.2. Gfun

generating functions package
KEYWORD: Symbolic computation
FUNCTIONAL DESCRIPTION: Gfun is a Maple package for the manipulation of linear recurrence or differential equations. It provides tools for guessing a sequence or a series from its first terms, for manipulating rigorously solutions of linear differential or recurrence equations, using the equation as a data-structure.

- Contact: Bruno Salvy
- URL: http://perso.ens-lyon.fr/bruno.salvy/software/the-gfun-package/

6.3. GNU-MPFR

KEYWORDS: Multiple-Precision - Floating-point - Correct Rounding
FUNCTIONAL DESCRIPTION: GNU MPFR is an efficient multiple-precision floating-point library with well-defined semantics (copying the good ideas from the IEEE-754 standard), in particular correct rounding in 5 rounding modes. GNU MPFR provides about 80 mathematical functions, in addition to utility functions (assignments, conversions...). Special data (Not a Number, infinities, signed zeros) are handled like in the IEEE-754 standard.

- Participants: Guillaume Hanrot, Paul Zimmermann, Philippe Théveny and Vincent Lefèvre
- Contact: Vincent Lefèvre
- URL: http://www.mpfr.org/

6.4. Sipe

KEYWORDS: Floating-point - Correct Rounding
FUNCTIONAL DESCRIPTION: Sipe is a mini-library in the form of a C header file, to perform radix-2 floating-point computations in very low precisions with correct rounding, either to nearest or toward zero. The goal of such a tool is to do proofs of algorithms/properties or computations of tight error bounds in these precisions by exhaustive tests, in order to try to generalize them to higher precisions. The currently supported operations are addition, subtraction, multiplication (possibly with the error term), fused multiply-add/subtract (FMA/FMS), and miscellaneous comparisons and conversions. Sipe provides two implementations of these operations, with the same API and the same behavior: one based on integer arithmetic, and a new one based on floating-point arithmetic.

- Participant: Vincent Lefèvre
- Contact: Vincent Lefèvre
- URL: https://www.vinc17.net/research/sipe/

6.5. LinBox

KEYWORD: Exact linear algebra
FUNCTIONAL DESCRIPTION: LinBox is an open-source C++ template library for exact, high-performance linear algebra computations. It is considered as the reference library for numerous computations (such as linear system solving, rank, characteristic polynomial, Smith normal forms,...) over finite fields and integers with dense, sparse, and structured matrices.

- Participants: Clément Pernet and Thierry Gautier
- Contact: Clément Pernet
- URL: http://linalg.org/

6.6. HPLLL

KEYWORDS: Computer algebra system (CAS) - Euclidean Lattices
FUNCTIONAL DESCRIPTION: Software library for linear algebra and Euclidean lattice problems

- Contact: Gilles Villard
- URL: http://perso.ens-lyon.fr/gilles.villard/hplll/

AROMATH Project-Team

5. New Software and Platforms

5.1. Platforms

5.1.1. Axel

KEYWORDS: Algorithm , CAD , Numerical algorithm , Geometric algorithms
SCIENTIFIC DESCRIPTION

Axel is an algebraic geometric modeler that aims at providing "algebraic modeling" tools for the manipulation and computation with curves, surfaces or volumes described by semi-algebraic representations. These include parametric and implicit representations of geometric objects. Axel also provides algorithms to compute intersection points or curves, singularities of algebraic curves or surfaces, certified topology of curves and surfaces, etc. A plugin mechanism allows to extend easily the data types and functions available in the plateform.
FUNCTIONAL DESCRIPTION

Axel is a cross platform software to visualize, manipulate and compute 3D objects. It is composed of a main application and several plugins. The main application provides atomic geometric data and processes, a viewer based on VTK, a GUI to handle objects, to select data, to apply process on them and to visualize the results. The plugins provides more data with their reader, writer, converter and interactors, more processes on the new or atomic data. It is written in C++ and thanks to a wrapping system using SWIG, its data structures and algorithms can be integrated into C# programs, as well as Python. The software is distributed as a source package, as well as binary packages for Linux, MacOSX and Windows.

- Participants: Nicolas Douillet, Anaïs Ducoffe, Valentin Michelet, Bernard Mourrain, Meriadeg Perrinel, Stéphane Chau and Julien Wintz
- Contact: Bernard Mourrain
- URL: http://axel.inria.fr/

Collaboration with Elisa Berrini (MyCFD, Sophia), Tor Dokken (Gotools library, Oslo, Norway), Angelos Mantzaflaris (GISMO library, Linz, Austria), Laura Saini (Post-Doc GALAAD/Missler, TopSolid), Gang Xu (Hangzhou Dianzi University, China), Meng Wu (Hefei University of Technology, China).

5.1.2. Dtk-Nurbs-Probing

KEYWORDS: CAO - Algebraic geometric modeler
SCIENTIFIC DESCRIPTION

This library offers tools for computing intersection between linear primitives and the constitutive elements of CAD objects (curves and surfaces). It is thus possible to compute intersections between a linear primitive with a trimmed NURBS surface, as well as untrimmed, moreover with a Bezier surface. It is also possible, in the xy plane, to compute the intersections between linear primitives and NURBS curves as well as Bezier curves.
FUNCTIONAL DESCRIPTION

Polynomial/rational defined primitives intersection with linear primitives under the form of a dtk plugin.

- Authors: Come Le Breton, Laurent Busé, Pierre Alliez, Julien Wintz, Thibaud Kloczko.
- Contact: Laurent Busé
- URL: http://nurbsprobing.inria.fr/

Collaboration with Pierre Alliez (Titane) and the industrial partner GeometryFactory (Sophia).

CARAMBA Project-Team

6. New Software and Platforms

6.1. Belenios

Belenios - Verifiable online voting system
KEYWORD: E-voting
FUNCTIONAL DESCRIPTION: Belenios is an online voting system that provides confidentiality and verifiability. End-to-end verifiablity relies on the fact that the ballot box is public (voters can check that their ballots have been received) and on the fact that the tally is publicly verifiable (anyone can recount the votes). Confidentiality relies on the encryption of the votes and the distribution of the decryption key.

Belenios builds upon Helios, a voting protocol used in several elections. The main design enhancement of Belenios vs Helios is that the ballot box can no longer add (fake) ballots, due to the use of credentials.

- Participants: Pierrick Gaudry, Stéphane Glondu and Véronique Cortier
- Partners: CNRS - Inria
- Contact: Stéphane Glondu
- URL: http://belenios.gforge.inria.fr/

6.2. tinygb

KEYWORD: Gröbner bases
FUNCTIONAL DESCRIPTION: Tinygb is a free software which implements tools for computing Gröbner bases with Faugère's F4 algorithm.
NEWS OF THE YEAR: The code has been largely rewritten and optimized. A new release is planned for the beginning of 2018.

- Author: Pierre-Jean Spaenlehauer
- Contact: Pierre-Jean Spaenlehauer
- URL: https://gforge.inria.fr/projects/tinygb/

6.3. CADO-NFS

Crible Algébrique: Distribution, Optimisation - Number Field Sieve
KEYWORDS: Cryptography - Number theory
FUNCTIONAL DESCRIPTION: CADO-NFS is a complete implementation in C/C++ of the Number Field Sieve (NFS) algorithm for factoring integers and computing discrete logarithms in finite fields. It consists in various programs corresponding to all the phases of the algorithm, and a general script that runs them, possibly in parallel over a network of computers.

- Participants: Pierrick Gaudry, Emmanuel Thomé and Paul Zimmermann
- Contact: Emmanuel Thomé
- URL: http://cado-nfs.gforge.inria.fr/

CASCADE Project-Team (section vide)

DATASHAPE Project-Team

6. New Software and Platforms

6.1. GUDHI

Geometric Understanding in Higher Dimensions
KEYWORDS: Computational geometry - Topology
SCIENTIFIC DESCRIPTION: The current release of the GUDHI library includes: – Data structures to represent, construct and manipulate simplicial and cubical complexes. – Algorithms to compute simplicial complexes from point cloud data. – Algorithms to compute persistent homology and multi-field persistent homology. – Simplification methods via implicit representations.
FUNCTIONAL DESCRIPTION: The GUDHI open source library will provide the central data structures and algorithms that underly applications in geometry understanding in higher dimensions. It is intended to both help the development of new algorithmic solutions inside and outside the project, and to facilitate the transfer of results in applied fields.

RELEASE FUNCTIONAL DESCRIPTION: Major new features in 2017: - python interface - bottleneck distance - tangential complex - relaxed witness complex

- Participants: Clément Maria, François Godi, David Salinas, Jean-Daniel Boissonnat, Marc Glisse, Mariette Yvinec, Pawel Dlotko, Siargey Kachanovich and Vincent Rouvreau
- Contact: Jean-Daniel Boissonnat
- URL: http://gudhi.gforge.inria.fr/

6.2. dD Triangulations

CGAL module: Triangulations in any dimension
KEYWORDS: 3D modeling - Triangulation - Delaunay triangulation - Voronoi diagram - Regular triangulation
FUNCTIONAL DESCRIPTION: This package of CGAL (Computational Geometry Algorithms Library http://www.cgal.org) allows to compute triangulations, Delaunay triangulations and regular triangulations in any dimension. Those triangulations are built incrementally and can be modified by insertion or removal of vertices.

RELEASE FUNCTIONAL DESCRIPTION: Version 4.11 adds the regular triangulations to the package.

- Participants: Clément Jamin, Olivier Devillers and Samuel Hornus
- Contact: Samuel Hornus
- URL: http://www.cgal.org

GAMBLE Project-Team

6. New Software and Platforms

6.1. ISOTOP

Topology and geometry of planar algebraic curves
KEYWORDS: Topology - Curve plotting - Geometric computing
FUNCTIONAL DESCRIPTION: Isotop is a Maple software for computing the topology of an algebraic plane curve, that is, for computing an arrangement of polylines isotopic to the input curve. This problem is a necessary key step for computing arrangements of algebraic curves and has also applications for curve plotting. This software has been developed since 2007 in collaboration with F. Rouillier from Inria Paris - Rocquencourt. It is based on the method described in [Cheng, J., Lazard, S., Pe
NEWS OF THE YEAR: In 2017, an ADT FastTrack funded a 6 months engineer contract to port the Maple code to C code. In addition, another local engineer from Inria Nancy (Benjamin Dexheimer) implemented a web server to improve the diffusion of our software.

- Participants: Elias Tsigaridas, Jinsan Cheng, Luis Penaranda, Marc Pouget and Sylvain Lazard
- Contact: Sylvain Lazard
- URL: http://vegas.loria.fr/isotop/

6.2. CGAL Package : 3D periodic regular triangulations

KEYWORDS: Flat torus - CGAL - Geometry - Geometric computing - Voronoi diagram - Delaunay triangulation - Triangulation
FUNCTIONAL DESCRIPTION: This class of CGAL (Computational Geometry Algorithms Library http://www.cgal.org) allows to build and handle periodic regular triangulations whose fundamental domain is a cube in 3D. Triangulations are built incrementally and can be modified by insertion of weighted points or removal of vertices. They offer location facilities for weighted points. The class offers nearest neighbor queries for the additively weighted distance and primitives to build the dual weighted Voronoi diagrams.

- Participants: Aymeric Pellé, Mael Rouxel-Labbe and Monique Teillaud
- Contact: Monique Teillaud
- URL: https://doc.cgal.org/latest/Manual/packages.html#PkgPeriodic3Triangulation3Summary

6.3. CGAL Package : 2D hyperbolic triangulations

KEYWORDS: Geometry - Delaunay triangulation - Hyperbolic space
FUNCTIONAL DESCRIPTION: This package implements the construction of Delaunay triangulations in the Poincaré disk model.

- Authors: Mikhail Bogdanov, Olivier Devillers and Monique Teillaud
- Contact: Monique Teillaud
- Publication: Hyperbolic Delaunay Complexes and Voronoi Diagrams Made Practical
- URL: https://github.com/CGAL/cgal-public-dev/tree/Hyperbolic_triangulation_2-MBogdanov

6.4. CGAL Package : 2D periodic hyperbolic triangulations

KEYWORDS: Geometry - Delaunay triangulation - Hyperbolic space
FUNCTIONAL DESCRIPTION: This module implements the computation of Delaunay triangulations of the Bolza surface.

- Authors: Iordan Iordanov and Monique Teillaud
- Contact: Monique Teillaud
- Publication: Implementing Delaunay Triangulations of the Bolza Surface
- URL: https://github.com/CGAL/cgal-public-dev/tree/Periodic_4_hyperbolic_triangulation_2-IIordanov

GRACE Project-Team

5. New Software and Platforms

5.1. ACTIS

Algorithmic Coding Theory in Sage
FUNCTIONAL DESCRIPTION: The aim of this project is to vastly improve the state of the error correcting library in Sage. The existing library does not present a good and usable API, and the provided algorithms are very basic, irrelevant, and outdated. We thus have two directions for improvement: renewing the APIs to make them actually usable by researchers, and incorporating efficient programs for decoding, like J. Nielsen's CodingLib, which contains many new algorithms.

- Partner: Technical University Denmark
- Contact: Daniel Augot

5.2. DECODING

KEYWORD: Algebraic decoding
FUNCTIONAL DESCRIPTION: Decoding is a standalone C library. Its primary goal is to implement Guruswami–Sudan list decoding-related algorithms, as efficiently as possible. Its secondary goal is to give an efficient tool for the implementation of decoding algorithms (not necessarily list decoding algorithms) and their benchmarking.

- Participant: Guillaume Quintin
- Contact: Daniel Augot

5.3. Fast Compact Diffie-Hellman

KEYWORD: Cryptography
FUNCTIONAL DESCRIPTION: A competitive, high-speed, open implementation of the Diffie–Hellman protocol, targeting the 128-bit security level on Intel platforms. This download contains Magma files that demonstrate how to compute scalar multiplications on the x-line of an elliptic curve using endomorphisms. This accompanies the EuroCrypt 2014 paper by Costello, Hisil and Smith, the full version of which can be found here: http://eprint.iacr.org/2013/692 . The corresponding SUPERCOP-compatible crypto_dh application can be downloaded from http://hhisil.yasar.edu.tr/files/hisil20140318compact.tar.gz .

- Participant: Benjamin Smith
- Contact: Benjamin Smith
- URL: http://research.microsoft.com/en-us/downloads/ef32422a-af38-4c83-a033-a7aafbc1db55/

5.4. CADO-NFS

Crible Algébrique: Distribution, Optimisation - Number Field Sieve
KEYWORDS: Cryptography - Number theory
FUNCTIONAL DESCRIPTION: CADO-NFS is a complete implementation in C/C++ of the Number Field Sieve (NFS) algorithm for factoring integers and computing discrete logarithms in finite fields. It consists in various programs corresponding to all the phases of the algorithm, and a general script that runs them, possibly in parallel over a network of computers.

- Participants: Pierrick Gaudry, Emmanuel Thomé and Paul Zimmermann
- Contact: Emmanuel Thomé
- URL: http://cado-nfs.gforge.inria.fr/

LFANT Project-Team

5. New Software and Platforms

5.1. APIP

Another Pairing Implementation in PARI
SCIENTIFIC DESCRIPTION: Apip , Another Pairing Implementation in PARI, is a library for computing standard and optimised variants of most cryptographic pairings.

The following pairings are available: Weil, Tate, ate and twisted ate, optimised versions (à la Vercauteren–Hess) of ate and twisted ate for selected curve families.

The following methods to compute the Miller part are implemented: standard Miller double-and-add method, standard Miller using a non-adjacent form, Boxall et al. version, Boxall et al. version using a non-adjacent form.

The final exponentiation part can be computed using one of the following variants: naive exponentiation, interleaved method, Avanzi–Mihailescu's method, Kato et al.'s method, Scott et al.'s method.

Part of the library has been included into Pari/Gp proper.
FUNCTIONAL DESCRIPTION: APIP is a library for computing standard and optimised variants of most cryptographic pairings.

- Participant: Jérôme Milan
- Contact: Jérôme Milan
- URL: http://www.lix.polytechnique.fr/~milanj/apip/apip.xhtml

5.2. AVIsogenies

Abelian Varieties and Isogenies
FUNCTIONAL DESCRIPTION: AVIsogenies is a Magma package for working with abelian varieties, with a particular emphasis on explicit isogeny computation.

Its prominent feature is the computation of (l,l)-isogenies between Jacobian varieties of genus-two hyperelliptic curves over finite fields of characteristic coprime to l, practical runs have used values of l in the hundreds.

It can also be used to compute endomorphism rings of abelian surfaces, and find complete addition laws on them.

- Participants: Damien Robert, Gaëtan Bisson and Romain Cosset
- Contact: Gaëtan Bisson
- URL: http://avisogenies.gforge.inria.fr/

5.3. CM

KEYWORD: Arithmetic
FUNCTIONAL DESCRIPTION: The Cm software implements the construction of ring class fields of imaginary quadratic number fields and of elliptic curves with complex multiplication via floating point approximations. It consists of libraries that can be called from within a C program and of executable command line applications.

RELEASE FUNCTIONAL DESCRIPTION: Features - Precisions beyond 300000 bits are now supported by an addition chain of variable length for the -function. Dependencies - The minimal version number of Mpfr has been increased to 3.0.0, that of Mpc to 1.0.0 and that of Pari to 2.7.0.

- Participant: Andreas Enge
- Contact: Andreas Enge
- URL: http://www.multiprecision.org/

5.4. CMH

Computation of Igusa Class Polynomials
KEYWORDS: Mathematics - Cryptography - Number theory
FUNCTIONAL DESCRIPTION: Cmh computes Igusa class polynomials, parameterising two-dimensional abelian varieties (or, equivalently, Jacobians of hyperelliptic curves of genus 2) with given complex multiplication.

- Participants: Andreas Enge, Emmanuel Thomé and Regis Dupont
- Contact: Emmanuel Thomé
- URL: http://cmh.gforge.inria.fr

5.5. CUBIC

FUNCTIONAL DESCRIPTION: Cubic is a stand-alone program that prints out generating equations for cubic fields of either signature and bounded discriminant. It depends on the Pari library. The algorithm has quasi-linear time complexity in the size of the output.

- Participant: Karim Belabas
- Contact: Karim Belabas
- URL: http://www.math.u-bordeaux1.fr/~belabas/research/software/cubic-1.2.tgz

5.6. Euclid

FUNCTIONAL DESCRIPTION: Euclid is a program to compute the Euclidean minimum of a number field. It is the practical implementation of the algorithm described in [38] . Some corresponding tables built with the algorithm are also available. Euclid is a stand-alone program depending on the PARI library.

- Participants: Jean-Paul Cerri and Pierre Lezowski
- Contact: Pierre Lezowski
- URL: http://www.math.u-bordeaux1.fr/~plezowsk/euclid/index.php

5.7. KleinianGroups

FUNCTIONAL DESCRIPTION: KleinianGroups is a Magma package that computes fundamental domains of arithmetic Kleinian groups.

- Participant: Aurel Page
- Contact: Aurel Page
- URL: http://www.normalesup.org/~page/Recherche/Logiciels/logiciels-en.html

5.8. GNU MPC

KEYWORD: Arithmetic
FUNCTIONAL DESCRIPTION: Mpc is a C library for the arithmetic of complex numbers with arbitrarily high precision and correct rounding of the result. It is built upon and follows the same principles as Mpfr. The library is written by Andreas Enge, Philippe Théveny and Paul Zimmermann.

RELEASE FUNCTIONAL DESCRIPTION: Fixed `mpc_pow`, see http://lists.gforge.inria.fr/pipermail/mpc-discuss/2014-October/001315.html - #18257: Switched to libtool 2.4.5.

- Participants: Andreas Enge, Mickaël Gastineau, Paul Zimmermann and Philippe Théveny
- Contact: Andreas Enge
- URL: http://www.multiprecision.org/

5.9. MPFRCX

KEYWORD: Arithmetic
FUNCTIONAL DESCRIPTION: Mpfrcx is a library for the arithmetic of univariate polynomials over arbitrary precision real (Mpfr) or complex (Mpc) numbers, without control on the rounding. For the time being, only the few functions needed to implement the floating point approach to complex multiplication are implemented. On the other hand, these comprise asymptotically fast multiplication routines such as Toom-Cook and the FFT.

RELEASE FUNCTIONAL DESCRIPTION: - new function `product_and_hecke` - improved memory consumption for unbalanced FFT multiplications

- Participant: Andreas Enge
- Contact: Andreas Enge
- URL: http://www.multiprecision.org/

5.10. PARI/GP

KEYWORD: Computational number theory
FUNCTIONAL DESCRIPTION: Pari/Gp is a widely used computer algebra system designed for fast computations in number theory (factorisation, algebraic number theory, elliptic curves, modular forms ...), but it also contains a large number of other useful functions to compute with mathematical entities such as matrices, polynomials, power series, algebraic numbers, etc., and many transcendental functions.

- Participants: Andreas Enge, Hamish Ivey-Law, Henri Cohen and Karim Belabas
- Partner: CNRS
- Contact: Karim Belabas
- URL: http://pari.math.u-bordeaux.fr/

POLSYS Project-Team

5. New Software and Platforms

5.1. Epsilon

FUNCTIONAL DESCRIPTION: Epsilon is a library of functions implemented in Maple and Java for polynomial elimination and decomposition with (geometric) applications.

- Contact: Dongming Wang
- URL: http://wang.cc4cm.org/epsilon/index.html

5.2. FGb

KEYWORDS: Gröbner bases - Nonlinear system - Computer algebra
FUNCTIONAL DESCRIPTION: FGb is a powerful software for computing Gröbner bases. It includes the new generation of algorihms for computing Gröbner bases polynomial systems (mainly the F4, F5 and FGLM algorithms). It is implemented in C/C++ (approximately 250000 lines), standalone servers are available on demand. Since 2006, FGb is dynamically linked with Maple software (version 11 and higher) and is part of the official distribution of this software.

- Participant: Jean Charles Faugere
- Contact: Jean-Charles Faugère
- URL: http://www-polsys.lip6.fr/~jcf/FGb/index.html

5.3. FGb Light

FUNCTIONAL DESCRIPTION: Gröbner basis computation modulo p (p is a prime integer of 16 bits).

- Participant: Jean-Charles Faugère
- Contact: Jean-Charles Faugère
- URL: http://www-polsys.lip6.fr/~jcf/FGb/index.html

5.4. GBLA

FUNCTIONAL DESCRIPTION: GBLA is an open source C library for linear algebra specialized for eliminating matrices generated during Gröbner basis computations in algorithms like F4 or F5.

- Contact: Jean-Charles Faugère
- URL: http://www-polsys.lip6.fr/~jcf/GBLA/index.html

5.5. HFEBoost

FUNCTIONAL DESCRIPTION: Public-key cryptography system enabling an authentification of dematerialized data.

- Authors: Jean-Charles Faugère and Ludovic Perret
- Partner: UPMC
- Contact: Jean-Charles Faugère
- URL: http://www-polsys.lip6.fr/Links/hfeboost.html

5.6. RAGlib

Real Algebraic Geometry library

FUNCTIONAL DESCRIPTION: RAGLib is a powerful library, written in Maple, dedicated to solving over the reals polynomial systems. It is based on the FGb library for computing Grobner bases. It provides functionalities for deciding the emptiness and/or computing sample points to real solution sets of polynomial systems of equations and inequalities. This library provides implementations of the state-of-the-art algorithms with the currently best known asymptotic complexity for those problems.

- Contact: Mohab Safey El Din
- URL: http://www-polsys.lip6.fr/~safey/RAGLib/

5.7. SLV

FUNCTIONAL DESCRIPTION: SLV is a software package in C that provides routines for isolating (and subsequently refine) the real roots of univariate polynomials with integer or rational coefficients based on subdivision algorithms and on the continued fraction expansion of real numbers. Special attention is given so that the package can handle polynomials that have degree several thousands and size of coefficients hundrends of Megabytes. Currently the code consists of approx. 5000 lines.

- Contact: Elias Tsigaridas
- URL: http://www-polsys.lip6.fr/~elias/soft

5.8. SPECTRA

Semidefinite Programming solved Exactly with Computational Tools of Real Algebra
KEYWORD: Linear Matrix Inequalities
FUNCTIONAL DESCRIPTION: SPECTRA is a Maple library devoted to solving exactly Semi-Definite Programs. It can handle rank constraints on the solution. It is based on the FGb library for computing Gröbner bases and provides either certified numerical approximations of the solutions or exact representations thereof.

- Contact: Mohab Safey El Din
- URL: http://homepages.laas.fr/henrion/software/spectra/

SECRET Project-Team

6. New Software and Platforms

6.1. CFS

FUNCTIONAL DESCRIPTION: Reference implementation of parallel CFS (reinforced version of the digital signature scheme CFS). Two variants are proposed, one with a « bit-packing » finite field arithmetic and an evolution with a « bit-slicing » finite-field arithmetic (collaboration with Peter Schwabe). For 80 bits of security the running time for producing one signature with the « bit-packing » variant is slightly above one second. This is high but was still the fastest so far. The evolution with the « bit-slicing » arithmetic produces the same signature in about 100 milliseconds.

- Participants: Grégory Landais and Nicolas Sendrier
- Contact: Nicolas Sendrier
- URL: https://gforge.inria.fr/projects/cfs-signature/

6.2. Collision Decoding

KEYWORDS: Algorithm - Binary linear code
FUNCTIONAL DESCRIPTION: Collision Decoding implements two variants of information set decoding : Stern-Dumer, and MMT. To our knowledge it is the best full-fledged open-source implementation of generic decoding of binary linear codes. It is the best generic attack against code-based cryptography.

- Participants: Grégory Landais and Nicolas Sendrier
- Contact: Nicolas Sendrier
- URL: https://gforge.inria.fr/projects/collision-dec/

6.3. ISDF

FUNCTIONAL DESCRIPTION: Implementation of the Stern-Dumer decoding algorithm, and of a varaint of the algorithm due to May, Meurer and Thomae.

- Participants: Grégory Landais and Nicolas Sendrier
- Contact: Anne Canteaut
- URL: https://gforge.inria.fr/projects/collision-dec/

SPECFUN Project-Team

5. New Software and Platforms

5.1. DynaMoW

Dynamic Mathematics on the Web
FUNCTIONAL DESCRIPTION: Programming tool for controlling the generation of mathematical websites that embed dynamical mathematical contents generated by computer-algebra calculations. Implemented in OCaml.

- Participants: Alexis Darrasse, Frédéric Chyzak and Maxence Guesdon
- Contact: Frédéric Chyzak
- URL: http://ddmf.msr-inria.inria.fr/DynaMoW/

5.2. ECS

Encyclopedia of Combinatorial Structures
FUNCTIONAL DESCRIPTION: On-line mathematical encyclopedia with an emphasis on sequences that arise in the context of decomposable combinatorial structures, with the possibility to search by the first terms in the sequence, keyword, generating function, or closed form.

- Participants: Alexis Darrasse, Frédéric Chyzak, Maxence Guesdon and Stéphanie Petit
- Contact: Frédéric Chyzak
- URL: http://ecs.inria.fr/

5.3. DDMF

Dynamic Dictionary of Mathematical Functions
FUNCTIONAL DESCRIPTION: Web site consisting of interactive tables of mathematical formulas on elementary and special functions. The formulas are automatically generated by OCaml and computer-algebra routines. Users can ask for more terms of the expansions, more digits of the numerical values, proofs of some of the formulas, etc.

- Participants: Alexandre Benoit, Alexis Darrasse, Bruno Salvy, Christoph Koutschan, Frédéric Chyzak, Marc Mezzarobba, Maxence Guesdon, Stefan Gerhold and Thomas Gregoire
- Contact: Frédéric Chyzak
- URL: http://ddmf.msr-inria.inria.fr/1.9.1/ddmf

5.4. Mgfun

multivariate generating functions package
FUNCTIONAL DESCRIPTION: The Mgfun Project is a collection of packages for the computer algebra system Maple, and is intended for the symbolic manipulation of a large class of special functions and combinatorial sequences (in one or several variables and indices) that appear in many branches of mathematics, mathematical physics, and engineering sciences. Members of the class satisfy a crucial finiteness property which makes the class amenable to computer algebra methods and enjoy numerous algorithmic closure properties, including algorithmic closures under integration and summation.

- Contact: Frédéric Chyzak
- URL: http://specfun.inria.fr/chyzak/mgfun.html

5.5. Ssreflect

FUNCTIONAL DESCRIPTION: Ssreflect is a tactic language extension to the Coq system, developed by the Mathematical Components team.

- Participants: Assia Mahboubi, Cyril Cohen, Enrico Tassi, Georges Gonthier, Laurence Rideau, Laurent Théry and Yves Bertot
- Contact: Yves Bertot
- URL: http://math-comp.github.io/math-comp/

5.6. Math-Components

Mathematical Components library
FUNCTIONAL DESCRIPTION: The Mathematical Components library is a set of Coq libraries that cover the mechanization of the proof of the Odd Order Theorem.

RELEASE FUNCTIONAL DESCRIPTION: The library includes 16 more theory files, covering in particular field and Galois theory, advanced character theory, and a construction of algebraic numbers.

- Participants: Alexey Solovyev, Andrea Asperti, Assia Mahboubi, Cyril Cohen, Enrico Tassi, François Garillot, Georges Gonthier, Ioana Pasca, Jeremy Avigad, Laurence Rideau, Laurent Théry, Russell O'Connor, Sidi Ould Biha, Stéphane Le Roux and Yves Bertot
- Contact: Assia Mahboubi
- URL: http://math-comp.github.io/math-comp/

5.7. CoqInterval

Interval package for Coq
KEYWORDS: Interval arithmetic - Coq
FUNCTIONAL DESCRIPTION: CoqInterval is a library for the proof assistant Coq.

It provides several tactics for proving theorems on enclosures of real-valued expressions. The proofs are performed by an interval kernel which relies on a computable formalization of floating-point arithmetic in Coq.

The Marelle team developed a formalization of rigorous polynomial approximation using Taylor models in Coq. In 2014, this library has been included in CoqInterval.

- Participants: Assia Mahboubi, Érik Martin-Dorel, Guillaume Melquiond, Jean-Michel Muller, Laurence Rideau, Laurent Théry, Micaela Mayero, Mioara Joldes, Nicolas Brisebarre and Thomas Sibut-Pinote
- Contact: Guillaume Melquiond
- Publications: Proving bounds on real-valued functions with computations - Floating-point arithmetic in the Coq system - Proving Tight Bounds on Univariate Expressions with Elementary Functions in Coq - Formally Verified Approximations of Definite Integrals - Formally Verified Approximations of Definite Integrals
- URL: http://coq-interval.gforge.inria.fr/

CAIRN Project-Team

6. New Software and Platforms

6.1. Gecos

Generic Compiler Suite
KEYWORDS: Source-to-source compiler - Model-driven software engineering - Retargetable compilation
SCIENTIFIC DESCRIPTION: The Gecos (Generic Compiler Suite) project is a source-to-source compiler infrastructure developed in the Cairn group since 2004. It was designed to enable fast prototyping of program analysis and transformation for hardware synthesis and retargetable compilation domains.

Gecos is Java based and takes advantage of modern model driven software engineering practices. It uses the Eclipse Modeling Framework (EMF) as an underlying infrastructure and takes benefits of its features to make it easily extensible. Gecos is open-source and is hosted on the Inria gforge.

The Gecos infrastructure is still under very active development, and serves as a backbone infrastructure to projects of the group. Part of the framework is jointly developed with Colorado State University and between 2012 and 2015 it was used in the context of the FP7 ALMA European project. The Gecos infrastructure is currently used by the EMMTRIX start-up, a spin-off from the ALMA project which aims at commercializing the results of the project, and in the context of the H2020 ARGO European project.
FUNCTIONAL DESCRIPTION: GeCoS provides a programme transformation toolbox facilitating parallelisation of applications for heterogeneous multiprocessor embedded platforms. In addition to targeting programmable processors, GeCoS can regenerate optimised code for High Level Synthesis tools.

- Participants: Tomofumi Yuki, Thomas Lefeuvre, Imèn Fassi, Mickael Dardaillon, Ali Hassan El Moussawi and Steven Derrien
- Partner: Université de Rennes 1
- Contact: Steven Derrien
- URL: http://gecos.gforge.inria.fr

6.2. ID-Fix

Infrastructure for the Design of Fixed-point systems
KEYWORDS: Energy efficiency - Dynamic range evaluation - Accuracy optimization - Fixed-point arithmetic - Analytic Evaluation - Embedded systems - Code optimisation
SCIENTIFIC DESCRIPTION: The different techniques proposed by the team for fixed-point conversion are implemented on the ID.Fix infrastructure. The application is described with a C code using floating-point data types and different pragmas, used to specify parameters (dynamic, input/output word-length, delay operations) for the fixed-point conversion. This tool determines and optimizes the fixed-point specification and then, generates a C code using fixed-point data types (ac_fixed) from Mentor Graphics. The infrastructure is made-up of two main modules corresponding to the fixed-point conversion (ID.Fix-Conv) and the accuracy evaluation (ID.Fix-Eval)
FUNCTIONAL DESCRIPTION: ID.Fix focuses on computational precision accuracy and can provide an optimised specification using fixed point arithmetic from a C source code with floating point data types. Fixed point arithmetic is very widely used in embedded systems as it provides better performance and is much more energy efficient. ID.Fix used an analytic programme model which means it can explore more solutions and thereby produce much more efficient code.

- Participant: Olivier Sentieys
- Partner: Université de Rennes 1
- Contact: Olivier Sentieys
- URL: http://idfix.gforge.inria.fr

6.3. Platforms

6.3.1. Zyggie

KEYWORDS: Health - Biomechanics - Wireless body sensor networks - Low power - Gesture recognition - Hardware platform - Software platform - Localization

SCIENTIFIC DESCRIPTION: Zyggie is a hardware and software wireless body sensor network platform. Each sensor node, attached to different parts of the human body, contains inertial sensors (IMU) (accelerometer, gyrometer, compass and barometer), an embedded processor and a low-power radio module to communicate data to a coordinator node connected to a computer, tablet or smartphone. One of the system's key innovations is that it collects data from sensors as well as on distances estimated from the power of the radio signal received to make the 3D location of the nodes more precise and thus prevent IMU sensor drift and power consumption overhead. Zyggie can be used to determine posture or gestures and mainly has applications in sport, healthcare and the multimedia industry.

FUNCTIONAL DESCRIPTION: The Zyggie sensor platform was developed to create an autonomous Wireless Body Sensor Network (WBSN) with the capabilities of monitoring body movements. The Zyggie platform is part of the BoWI project funded by CominLabs. Zyggie is composed of a processor, a radio transceiver and different sensors including an Inertial Measurement Unit (IMU) with 3-axis accelerometer, gyrometer, and magnetometer. Zyggie is used for evaluating data fusion algorithms, low power computing algorithms, wireless protocols, and body channel characterization in the BoWI project.

The Zyggie V2 prototype includes the following features: a 32-bit microcontroller to manage a custom MAC layer and processe quaternions based on IMU measures, and an UWB radio from DecaWave to measure distances between nodes with Time of Flight (ToF).

- Participants: Arnaud Carer and Olivier Sentieys
- Partners: Lab-STICC - Université de Rennes 1
- Contact: Olivier Sentieys
- URL: http://www.bowi.cominlabs.ueb.eu/fr/zyggie-wbsn-platform

Figure 3. CAIRN*'s Ziggie platform for WBSN*

CAMUS Team

6. New Software and Platforms

6.1. APOLLO

Automatic speculative POLyhedral Loop Optimizer
KEYWORD: Automatic parallelization
FUNCTIONAL DESCRIPTION: APOLLO is dedicated to automatic, dynamic and speculative parallelization of loop nests that cannot be handled efficiently at compile-time. It is composed of a static part consisting of specific passes in the LLVM compiler suite, plus a modified Clang frontend, and a dynamic part consisting of a runtime system. It can apply on-the-fly any kind of polyhedral transformations, including tiling, and can handle nonlinear loops, as while-loops referencing memory through pointers and indirections.

- Participants: Aravind Sukumaran-Rajam, Juan Manuel Martinez Caamaño, Manuel Selva and Philippe Clauss
- Contact: Philippe Clauss
- URL: http://apollo.gforge.inria.fr

6.2. Clan

A Polyhedral Representation Extraction Tool for C-Based High Level Languages
KEYWORD: Polyhedral compilation
FUNCTIONAL DESCRIPTION: Clan is a free software and library which translates some particular parts of high level programs written in C, C++ or Java into a polyhedral representation called OpenScop. This representation may be manipulated by other tools to, e.g., achieve complex analyses or program restructurations (for optimization, parallelization or any other kind of manipulation). It has been created to avoid tedious and error-prone input file writing for polyhedral tools (such as CLooG, LeTSeE, Candl etc.). Using Clan, the user has to deal with source codes based on C grammar only (as C, C++ or Java). Clan is notably the frontend of the two major high-level compilers Pluto and PoCC.

- Participants: Cédric Bastoul and Imèn Fassi
- Contact: Cédric Bastoul
- URL: http://icps.u-strasbg.fr/people/bastoul/public_html/development/clan/

6.3. Clay

Chunky Loop Alteration wizardrY
FUNCTIONAL DESCRIPTION: Clay is a free software and library devoted to semi-automatic optimization using the polyhedral model. It can input a high-level program or its polyhedral representation and transform it according to a transformation script. Classic loop transformations primitives are provided. Clay is able to check for the legality of the complete sequence of transformation and to suggest corrections to the user if the original semantics is not preserved.

- Participant: Cédric Bastoul
- Contact: Cédric Bastoul
- URL: http://icps.u-strasbg.fr/people/bastoul/public_html/development/clay/

6.4. CLooG

Code Generator in the Polyhedral Model

FUNCTIONAL DESCRIPTION: CLooG is a free software and library to generate code (or an abstract syntax tree of a code) for scanning Z-polyhedra. That is, it finds a code (e.g. in C, FORTRAN...) that reaches each integral point of one or more parameterized polyhedra. CLooG has been originally written to solve the code generation problem for optimizing compilers based on the polyhedral model. Nevertheless it is used now in various area e.g. to build control automata for high-level synthesis or to find the best polynomial approximation of a function. CLooG may help in any situation where scanning polyhedra matters. While the user has full control on generated code quality, CLooG is designed to avoid control overhead and to produce a very effective code. CLooG is widely used (including by GCC and LLVM compilers), disseminated (it is installed by default by the main Linux distributions) and considered as the state of the art in polyhedral code generation.

RELEASE FUNCTIONAL DESCRIPTION: It mostly solves building and offers a better OpenScop support.

- Participant: Cédric Bastoul
- Contact: Cédric Bastoul
- URL: http://www.cloog.org

6.5. IBB

Iterate-But-Better
FUNCTIONAL DESCRIPTION: IBB is a source-to-source xfor compiler which automatically translates any C source code containing xfor-loops into an equivalent source code where xfor-loops have been transformed into equivalent for-loops.

RELEASE FUNCTIONAL DESCRIPTION: The IBB compiler has been improved in some aspects in 2014: loop bounds can now be min and max functions, IBB uses the OpenScop format to encode statements and iteration domains.

- Participants: Cédric Bastoul, Imèn Fassi and Philippe Clauss
- Contact: Philippe Clauss
- URL: http://xfor.gforge.inria.fr

6.6. OpenScop

A Specification and a Library for Data Exchange in Polyhedral Compilation Tools
FUNCTIONAL DESCRIPTION: OpenScop is an open specification that defines a file format and a set of data structures to represent a static control part (SCoP for short), i.e., a program part that can be represented in the polyhedral model. The goal of OpenScop is to provide a common interface to the different polyhedral compilation tools in order to simplify their interaction. To help the tool developers to adopt this specification, OpenScop comes with an example library (under 3-clause BSD license) that provides an implementation of the most important functionalities necessary to work with OpenScop.

- Participant: Cédric Bastoul
- Contact: Cédric Bastoul
- URL: http://icps.u-strasbg.fr/people/bastoul/public_html/development/openscop/

6.7. PolyLib

The Polyhedral Library
KEYWORDS: Rational polyhedra - Library - Polyhedral compilation
SCIENTIFIC DESCRIPTION: A C library used in polyhedral compilation, as a basic tool used to analyze, transform, optimize polyhedral loop nests. Has been shipped in the polyhedral tools Cloog and Pluto.

FUNCTIONAL DESCRIPTION: PolyLib is a C library of polyhedral functions, that can manipulate unions of rational polyhedra of any dimension. It was the first to provide an implementation of the computation of parametric vertices of a parametric polyhedron, and the computation of an Ehrhart polynomial (expressing the number of integer points contained in a parametric polytope) based on an interpolation method. Vincent Loechner is the maintainer of this software.

- Participant: Vincent Loechner
- Contact: Vincent Loechner
- URL: http://icps.u-strasbg.fr/PolyLib/

6.8. ORWL

Ordered Read-Write Lock
KEYWORDS: Task scheduling - Deadlock detection
FUNCTIONAL DESCRIPTION: ORWL is a reference implementation of the Ordered Read-Write Lock tools. The macro definitions and tools for programming in C99 that have been implemented for ORWL have been separated out into a toolbox called P99.

- Participants: Jens Gustedt, Mariem Saied and Stéphane Vialle
- Contact: Jens Gustedt
- Publications: Iterative Computations with Ordered Read-Write Locks - Automatic, Abstracted and Portable Topology-Aware Thread Placement - Resource-Centered Distributed Processing of Large Histopathology Images - Automatic Code Generation for Iterative Multi-dimensional Stencil Computations

6.9. P99

KEYWORD: Macro programming
FUNCTIONAL DESCRIPTION: P99 is a suite of macro and function definitions that ease the programming in modern C, minimum C99. By using tools from C99 and C11 we implement default arguments for functions, scope bound resource management, transparent allocation and initialization.

- Participants: Jens Gustedt, Mariem Saied and Stéphane Vialle
- Contact: Jens Gustedt
- URL: https://gforge.inria.fr/projects/p99/

6.10. stdatomic

standard atomic library
KEYWORD: Atomic access
SCIENTIFIC DESCRIPTION: We present a new algorithm and implementation of a lock primitive that is based on Linux' native lock interface, the futex system call. It allows us to assemble compiler support for atomic data structures that can not be handled through specific hardware instructions. Such a tool is needed for C11's atomics interface because here an _Atomic qualification can be attached to almost any data type. Our lock data structure for that purpose meets very specific criteria concerning its field of operation and its performance. By that we are able to outperform gcc's libatomic library by around 60%.
FUNCTIONAL DESCRIPTION: This implementation builds entirely on the two gcc ABIs for atomics. It doesn't even attempt to go down to assembly level by itself. We provide all function interfaces that the two gcc ABIs and the C standard need. For compilers that don't offer the direct language support for atomics this provides a syntactically reduced but fully functional approach to atomic operations.

- Author: Jens Gustedt
- Contact: Jens Gustedt
- Publications: Futex based locks for C11's generic atomics - Futex based locks for C11's generic atomics (extended abstract)
- URL: http://stdatomic.gforge.inria.fr/

6.11. musl

KEYWORDS: Standards - Library
SCIENTIFIC DESCRIPTION: musl provides consistent quality and implementation behavior from tiny embedded systems to full-fledged servers. Minimal machine-specific code means less chance of breakage on minority architectures and better success with "write once run everywhere" C development.

musl's efficiency is unparalleled in Linux libc implementations. Designed from the ground up for static linking, musl carefully avoids pulling in large amounts of code or data that the application will not use. Dynamic linking is also efficient, by integrating the entire standard library implementation, including threads, math, and even the dynamic linker itself into a single shared object, most of the startup time and memory overhead of dynamic linking have been eliminated.
FUNCTIONAL DESCRIPTION: We participate in the development of musl, a re-implementation of the C library as it is described by the C and POSIX standards. It is lightweight, fast, simple, free, and strives to be correct in the sense of standards-conformance and safety. Musl is production quality code that is mainly used in the area of embedded device. It gains more market share also in other area, e.g. there are now Linux distributions that are based on musl instead of Gnu LibC.

- Participant: Jens Gustedt
- Contact: Jens Gustedt
- URL: http://www.musl-libc.org/

6.12. Modular C

KEYWORDS: Programming language - Modularity
FUNCTIONAL DESCRIPTION: The change to the C language is minimal since we only add one feature, composed identifiers, to the core language. Our modules can import other modules as long as the import relation remains acyclic and a module can refer to its own identifiers and those of the imported modules through freely chosen abbreviations. Other than traditional C include, our import directive ensures complete encapsulation between modules. The abbreviation scheme allows to seamlessly replace an imported module by another one with equivalent interface. In addition to the export of symbols, we provide parameterized code injection through the import of "snippets". This implements a mechanism that allows for code reuse, similar to X macros or templates. Additional features of our proposal are a simple dynamic module initialization scheme, a structured approach to the C library and a migration path for existing software projects.

- Author: Jens Gustedt
- Contact: Jens Gustedt
- Publications: Modular C - Arbogast: Higher order AD for special functions with Modular C - Futex based locks for C11's generic atomics
- URL: http://cmod.gforge.inria.fr/

6.13. arbogast

KEYWORD: Automatic differentiation
SCIENTIFIC DESCRIPTION: This high-level toolbox for the calculus with Taylor polynomials is named after L.F.A. Arbogast (1759-1803), a French mathematician from Strasbourg (Alsace), for his pioneering work in derivation calculus. Its modular structure ensures unmatched efficiency for computing higher order Taylor polynomials. In particular it permits compilers to apply sophisticated vector parallelization to the derivation of nearly unmodified application code.

FUNCTIONAL DESCRIPTION: Arbogast is based on a well-defined extension of the C programming language, Modular C, and places itself between tools that proceed by operator overloading on one side and by rewriting, on the other. The approach is best described as contextualization of C code because it permits the programmer to place his code in different contexts – usual math or AD – to reinterpret it as a usual C function or as a differential operator. Because of the type generic features of modern C, all specializations can be delegated to the compiler.

- Author: Jens Gustedt
- Contact: Jens Gustedt
- Publications: Arbogast: Higher order AD for special functions with Modular C - Arbogast – Origine d'un outil de dérivation automatique
- URL: https://gforge.inria.fr/projects/arbo

6.14. CFML

Interactive program verification using characteristic formulae
KEYWORDS: Coq - Software Verification - Deductive program verification - Separation Logic
FUNCTIONAL DESCRIPTION: The CFML tool supports the verification of OCaml programs through interactive Coq proofs. CFML proofs establish the full functional correctness of the code with respect to a specification. They may also be used to formally establish bounds on the asymptotic complexity of the code. The tool is made of two parts: on the one hand, a characteristic formula generator implemented as an OCaml program that parses OCaml code and produces Coq formulae, and, on the other hand, a Coq library that provides notation and tactics for manipulating characteristic formulae interactively in Coq.

- Participants: Arthur Charguéraud, Armaël Guéneau and François Pottier
- Contact: Arthur Charguéraud
- URL: http://www.charguéraud.org/softs/cfml/

6.15. TLC

TLC Coq library
KEYWORDS: Coq - Library
FUNCTIONAL DESCRIPTION: TLC is a general purpose Coq library that provides an alternative to Coq's standard library. TLC takes as axiom extensionality, classical logic and indefinite description (Hilbert's epsilon). These axioms allow for significantly simpler formal definitions in many cases. TLC takes advantage of the type class mechanism. In particular, this allows for common operators and lemma names for all container data structures and all order relations. TLC includes the optimal fixed point combinator, which can be used for building arbitrarily-complex recursive and co-recursive definitions. Last, TLC provides a collection of tactics that enhance the default tactics provided by Coq. These tactics help constructing more concise and more robust proof scripts.

- Contact: Arthur Charguéraud
- URL: http://www.charguéraud.org/softs/tlc/

CORSE Project-Team

5. New Software and Platforms

5.1. THEMIS

THEMIS: A Tool for Decentralized Monitoring Algorithms
KEYWORDS: Monitoring - Simulation
FUNCTIONAL DESCRIPTION: THEMIS consists of a library and command-line tools. It provides an API, data structures and measures for decentralized monitoring. These building blocks can be reused or extended to modify existing algorithms, design new more intricate algorithms, and elaborate new approaches to assess existing algorithms.

- Participants: Antoine El Hokayem and Ylies Falcone
- Contact: Antoine El Hokayem
- Publications: THEMIS: A Tool for Decentralized Monitoring Algorithms - Monitoring Decentralized Specifications
- URL: https://gitlab.inria.fr/monitoring/themis/

5.2. Verde

KEYWORDS: Debug - Verification
FUNCTIONAL DESCRIPTION: Interactive Debugging with a traditional debugger can be tedious. One has to manually run a program step by step and set breakpoints to track a bug.

i-RV is an approach to bug fixing that aims to help developpers during their Interactive Debbugging sessions using Runtime Verification.

Verde is the reference implementation of i-RV.

- Participants: Kevin Pouget, Ylies Falcone, Raphael Jakse and Jean-François Méhaut
- Contact: Raphael Jakse
- Publication: Interactive Runtime Verification - When Interactive Debugging meets Runtime Verification
- URL: https://gitlab.inria.fr/monitoring/verde

5.3. Nanvix

KEYWORD: Operating system
SCIENTIFIC DESCRIPTION: Nanvix presents a similar structure to Unix System V , and it has been intentionally designed this way because it is adopted in some successful Operating Systemes, such as Linux. Nanvix is structured in two layers. The kernel (bottom layer), seats on top of the hardware and runs in privileged mode. Its job is to (i) extend the underlying hardware so that an easier-to-program interface is exported to the higher layer, and (ii) multiplex hardware resources among several users. The userland (top layer), relies on Posix system calls exported by the kernel and it is the place where user software run in unprivileged mode.

The kernel presents a tiny monolithic architecture (7k loc), and it is structured in four subsystems: the hardware abstraction layer, the memory management system, the process manager, and the file system. The hardware abstraction layer interacts directly with the hardware and exports to the other subsystems a set of well-defined low-level routines. The job of the hardware abstraction layer is to isolate, as much as possible, all the hardware intricacies, so that the kernel can easily be ported to other compatible platforms.

The memory manager provides a flat virtual memory abstraction. It does so by having two modules working together: the paging and virtual memory allocator. The former deals with paging, keeping in memory those pages that are more frequently used, and swapping out to disk those that are not. The virtual memory allocator, on the other hand, relies on the paging module to create higher-level abstractions called memory regions, and thus enable advanced features such as shared memory regions, on-demand loading and lazy coping.

The process manager handles creation, termination, scheduling, synchronization and communication of processes. Processes are single-threaded entities and are created on demand, either by the system itself or the user. Scheduling is based on preemption, and in userland it happens whenever a process runs out of quantum or blocks awaiting for a resource. In kernel land, processes run in nonpreemptive mode and scheduling occurs when a processes voluntarily foes to sleep. In addition, the process manager exports inter-process communication facilities, such as Posix pipes and shared memory regions.

The file system provides a uniform interface for dealing with hardware resources. It extends the device driver interface and creates on top of it the file abstraction. Files can be accessed through a unique pathname, and may be shared among several processes. The Nanvix file system is compatible with the one present in Minix, it adopts an hierarchical inode structure, and features mounting points and disk block caching.

Investigations on Nanvix concern to a joint collaboration research effort between the CORSE Team (Inria - FRANCE) and CArT (PUC Minas - Brazil). More precisely, a port of Nanvix to low-power embedded many-cores is ongoing, and it consists on the thesis subject of a cotutella student between the two aforementioned research teams.
FUNCTIONAL DESCRIPTION: Nanvix is an Operating System that we designed from scratch to address growing interested on research and education. It originally targets x86-based architectures and features virtual-memory based on paging, a hierarchical Unix file system based on inodes, a uniform device driver interface, and a preemptive priority-based scheduler.

We are currently extending Nanvix to provide a portable OS targetting multiple manycore platforms through the PhD of Pedro Henrique Penna.

- Participants: Pedro Henrique De Mello Morado Penna, François Broquedis, Jean-François Méhaut, Marcio Bastos Castro and Henrique Cota De Freitas
- Partner: Université pontificale catholique du Minas Gerais
- Contact: Pedro Henrique De Mello Morado Penna
- URL: https://github.com/nanvix/nanvix

5.4. Mickey

KEYWORDS: Dynamic Analysis - Performance analysis - Profiling - Polyhedral compilation
FUNCTIONAL DESCRIPTION: Mickey is a set of tools for profiling based performance debugging for compiled binaries. It uses a dynamic binary translator to instrument arbitrary programs as they are being run to reconstruct the control flow and track data dependencies. This information is then fed to a polyhedral optimizer that proposes structured transformations for the original code.

Mickey can handle both inter- and intra-procedural control and data flow in a unified way, thus enabling inter-procedural structured transformations. It is based on QEMU to allow for portability, both in terms of targeted CPU architectures, but also in terms of programming environment and the use of third-party libraries for which no source code is available.

- Partner: STMicroelectronics
- Contact: Fabian Gruber

5.5. IPFME

Integer Polynomial Fourier-Motzkin Elimination

KEYWORDS: Fourier–Motzkin Elimination - Quantifier Elimination - System of Inequalities - Mixed Integer Programming - Polynomial or analytical systems
SCIENTIFIC DESCRIPTION: Fourier-Motzkin is a very well known algorithm for performing quantifier (variable) elimination, given a system (or formula) of inequalities. It removes quantified variables by combining all upper and lower bounds of such variables.

It was designed to operate on linear systems, where all coefficients of the variable being eliminated are numeric values, and the inequality can be classified as either a upper or lower bound.

When dealing with polynomials, variable coefficients might be symbolic expressions. In such case, all possible signs of the coefficient (positive, negative, or zero) must be explored.

To avoid this branching we use the positiveness test algorithm, proposed by Markus Schweighofer (https://doi.org/10.1016/S0022-4049(01)00041-X), to retrieve symbolic coefficient signs.

The same positiveness test algorithm is of major importance when resolving system over integer variables, instead of reals. It is used in many other techniques required to preserve the precision of the simplified formula, such as extending the normalization technique (https://doi.org/10.1145/125826.125848) to symbolic expressions, performing convex hull detection and removing redundant constraints. Such tester is implemented using GLPK (https://www.gnu.org/software/glpk).
FUNCTIONAL DESCRIPTION: Quantifier elimination is the process of removing existential variables of a given formula, obtaining one with less variables and that implies the original formula. This can also be viewed as a projection of the set of points (integer here) that satisfy the original formula onto a sub-vectorial space made up of all the non-eliminated variables. The obtained projection is an over-approximation of the exact projection. The goal of the process is to make it as tight as possible.

IPFME presents extensions to the Fourier-Motzkin quantifier elimination process. The developed techniques allow to derive more precise simplification operations when handling integer valued multivariate polynomial systems.

The implementation, in C++, uses GiNaC (https://www.ginac.de/) for the manipulation of symbolic expressions.

- Authors: Diogo Nunes Sampaio, Fabrice Rastello and Alain Ketterlin
- Contact: Diogo Nunes Sampaio
- Publications: Profile Guided Hybrid Compilation - Simplification and Run-time Resolution of Data Dependence Constraints for Loop Transformations

5.6. mcGDB

Model Centric Debugging with GDB
KEYWORDS: Model debugging - Parallel programming - OpenMP - Multicore
FUNCTIONAL DESCRIPTION: mcGDB defines the concept of "programming-model centric" source-level interactive debugging as an extension of the traditional language-level interactive debugging. The idea is to integrate into debuggers the notion of "programming models", as abstract machines running over the physical ones. These abstract machines, implemented by runtime libraries and programming frameworks, provide high-level primitives required for the implementation of today's parallel applications. mcGDB is developed as a Python extension of GDB, the debugger of the GNU project

- Partner: STMicroelectronics
- Contact: Jean-François Méhaut
- URL: http://dema.gforge.inria.fr/

5.7. BOAST

Bringing Optimization Through Automatic Source-to-Source Transformations

KEYWORDS: Code generation - Portability - Autotuning - High performance computing - Conformance testing - Productivity
FUNCTIONAL DESCRIPTION: BOAST provides scientific application developers with a framework to develop and test application computing kernels.

The developer starts from an application kernel (either designed or implemented), and writes it in a dedicated language. This language provides enough flexibility for the kernel to be metaprogrammed with several orthogonal optimizations. From this set of optimizations, possible languages targets, and compilation options, the user can design an optimization space to explore. This optimization space can contain rules to remove infeasible candidates. BOAST provides the mechanisms to specify those optimization spaces and enforce the users rules.

BOAST was already used with three real scientific applications: BigDFT (materials, CEA Inac), SPECFEM3D (geophysics, CNRS and Princeton) and GYSELA (plasma physics, CEA Cadarache, ITER). '

- Partner: CEA INAC LSim
- Contact: Brice Videau
- URL: https://github.com/Nanosim-LIG/boast

PACAP Project-Team

5. New Software and Platforms

5.1. ATMI

KEYWORDS: Analytic model - Chip design - Temperature
SCIENTIFIC DESCRIPTION: Research on temperature-aware computer architecture requires a chip temperature model. General purpose models based on classical numerical methods like finite differences or finite elements are not appropriate for such research, because they are generally too slow for modeling the time-varying thermal behavior of a processing chip.

We have developed an ad hoc temperature model, ATMI (Analytical model of Temperature in MIcroprocessors), for studying thermal behaviors over a time scale ranging from microseconds to several minutes. ATMI is based on an explicit solution to the heat equation and on the principle of superposition. ATMI can model any power density map that can be described as a superposition of rectangle sources, which is appropriate for modeling the microarchitectural units of a microprocessor.
FUNCTIONAL DESCRIPTION: ATMI is a library for modelling steady-state and time-varying temperature in microprocessors. ATMI uses a simplified representation of microprocessor packaging.

- Participant: Pierre Michaud
- Contact: Pierre Michaud
- URL: https://team.inria.fr/pacap/software/atmi/

5.2. HEPTANE

KEYWORDS: IPET - WCET - Performance - Real time - Static analysis - Worst Case Execution Time
SCIENTIFIC DESCRIPTION: WCET estimation

Status: Registered with APP (Agence de Protection des Programmes). Available under GNU General Public License v3, with number IDDN.FR.001.510039.000.S.P.2003.000.10600.

The aim of Heptane is to produce upper bounds of the execution times of applications. It is targeted at applications with hard real-time requirements (automotive, railway, aerospace domains). Heptane computes WCETs using static analysis at the binary code level. It includes static analyses of microarchitectural elements such as caches and cache hierarchies.

For more information, please contact Damien Hardy or Isabelle Puaut.
FUNCTIONAL DESCRIPTION: In a hard real-time system, it is essential to comply with timing constraints, and Worst Case Execution Time (WCET) in particular. Timing analysis is performed at two levels: analysis of the WCET for each task in isolation taking account of the hardware architecture, and schedulability analysis of all the tasks in the system. Heptane is a static WCET analyser designed to address the first issue.

- Participants: Benjamin Lesage, Loïc Besnard, Damien Hardy, François Joulaud, Isabelle Puaut and Thomas Piquet
- Partner: Université de Rennes 1
- Contact: Isabelle Puaut
- URL: https://team.inria.fr/pacap/software/heptane/

5.3. tiptop

KEYWORDS: Instructions - Cycles - Cache - CPU - Performance - HPC - Branch predictor
SCIENTIFIC DESCRIPTION: Tiptop is written in C. It can take advantage of libncurses when available for pseudo-graphic display.

Performance, hardware counters, analysis tool.

Status: Registered with APP (Agence de Protection des Programmes). Available under GNU General Public License v2, with number IDDN.FR.001.450006.000.S.P.2011.000.10800. Current version is 2.3.1, released October 2017.

Tiptop has been integrated in major Linux distributions, such as Fedora, Debian, Ubuntu.

Tiptop is a new simple and flexible user-level tool that collects hardware counter data on Linux platforms (version 2.6.31+). The goal is to make the collection of performance and bottleneck data as simple as possible, including simple installation and usage. In particular, we stress the following points.

Installation is only a matter of compiling the source code. No patching of the Linux kernel is needed, and no special-purpose module needs to be loaded.

No privilege is required, any user can run tiptop
FUNCTIONAL DESCRIPTION: Today's microprocessors have become extremely complex. To better understand the multitude of internal events, manufacturers have integrated many monitoring counters. Tiptop can be used to collect and display the values from these performance counters very easily. Tiptop may be of interest to anyone who wants to optimise the performance of their HPC applications.

- Participant: Erven Rohou
- Contact: Erven Rohou
- URL: http://tiptop.gforge.inria.fr

5.4. PADRONE

KEYWORDS: Legacy code - Optimization - Performance analysis - Dynamic Optimization
FUNCTIONAL DESCRIPTION: Padrone is new platform for dynamic binary analysis and optimization. It provides an API to help clients design and develop analysis and optimization tools for binary executables. Padrone attaches to running applications, only needing the executable binary in memory. No source code or debug information is needed. No application restart is needed either. This is especially interesting for legacy or commercial applications, but also in the context of cloud deployment, where actual hardware is unknown, and other applications competing for hardware resources can vary. The profiling overhead is minimum.

- Participants: Emmanuel Riou and Erven Rohou
- Contact: Erven Rohou
- URL: https://team.inria.fr/alf/software/padrone

5.5. If-memo

KEYWORD: Performance
SCIENTIFIC DESCRIPTION: We propose a linker based technique for enabling software memorizing of any dynamically linked pure function by function interception and we illustrate our framework using a set of computationally expensive pure functions - the transcendental functions. Our technique does not need the availability of source code and thus can even be applied to commercial applications as well as applications with legacy codes. As far as users are concerned, enabling memoization is as simple as setting an environment variable. Our framework does not make any specific assumptions about the underlying architecture or compiler too-chains, and can work with a variety of current architectures.

- Participants: Arjun Suresh and Erven Rohou
- Contact: Erven Rohou
- URL: https://team.inria.fr/alf/software/if-memo/

5.6. Simty

KEYWORDS: RISC-V - Multi-threading - SIMT - FPGA - Softcore - GPU

FUNCTIONAL DESCRIPTION: Simty is a massively multi-threaded processor core that dynamically assembles SIMD instructions from scalar multi-thread code. It runs the RISC-V (RV32-I) instruction set. Unlike existing SIMD or SIMT processors like GPUs, Simty takes binaries compiled for general-purpose processors without any instruction set extension or compiler changes. Simty is described in synthesizable VHDL.

- Author: Sylvain Collange
- Contact: Sylvain Collange
- URL: https://gforge.inria.fr/projects/simty

5.7. Barra

KEYWORDS: Performance - Computer architecture - Debug - Tesla ISA - GPU - Profiling - CUDA - HPC - Simulator - GPGPU
SCIENTIFIC DESCRIPTION: Research on throughout-oriented architectures demands accurate and representative models of GPU architectures in order to be able to evaluate new architectural ideas, explore design spaces and characterize applications. The Barra project is a simulator of the NVIDIA Tesla GPU architecture.

Barra builds upon knowledge acquired through micro-benchmarking, in order to provide a baseline model representative of industry practice. The simulator provides detailed statistics to identify optimization opportunities and is fully customizable to experiment ideas of architectural modifications. Barra incorporates both a functional model and a cycle-level performance model.
FUNCTIONAL DESCRIPTION: Barra is a Graphics Processing Unit (GPU) architecture simulator. It simulates NVIDIA CUDA programs at the assembly language level. Barra is a tool for research on computer architecture, and can also be used to debug, profile and optimize CUDA programs at the lowest level.

RELEASE FUNCTIONAL DESCRIPTION: Timing model Tesla-like architecture model Fermi-like architecture model New per-PC control-flow divergence management Simultaneous branch and warp interweaving Affine vector cache

- Participants: Alexandre Kouyoumdjian, David Defour, Fabrice Mouhartem and Sylvain Collange
- Partners: ENS Lyon - UPVD
- Contact: Sylvain Collange
- URL: http://barra.gforge.inria.fr/

AOSTE2 Team

6. New Software and Platforms

6.1. SynDEx

KEYWORDS: Distributed - Optimization - Real time - Embedded systems - Scheduling analyses
SCIENTIFIC DESCRIPTION: SynDEx is a system level CAD software implementing the AAA methodology for rapid prototyping and for optimizing distributed real-time embedded applications. It is developed in OCaML.

Architectures are represented as graphical block diagrams composed of programmable (processors) and non-programmable (ASIC, FPGA) computing components, interconnected by communication media (shared memories, links and busses for message passing). In order to deal with heterogeneous architectures it may feature several components of the same kind but with different characteristics.

Two types of non-functional properties can be specified for each task of the algorithm graph. First, a period that does not depend on the hardware architecture. Second, real-time features that depend on the different types of hardware components, ranging amongst execution and data transfer time, memory, etc.. Requirements are generally constraints on deadline equal to period, latency between any pair of tasks in the algorithm graph, dependence between tasks, etc.

Exploration of alternative allocations of the algorithm onto the architecture may be performed manually and/or automatically. The latter is achieved by performing real-time multiprocessor schedulability analyses and optimization heuristics based on the minimization of temporal or resource criteria. For example while satisfying deadline and latency constraints they can minimize the total execution time (makespan) of the application onto the given architecture, as well as the amount of memory. The results of each exploration is visualized as timing diagrams simulating the distributed real-time implementation.

Finally, real-time distributed embedded code can be automatically generated for dedicated distributed real-time executives, possibly calling services of resident real-time operating systems such as Linux/RTAI or Osek for instance. These executives are deadlock-free, based on off-line scheduling policies. Dedicated executives induce minimal overhead, and are built from processor-dependent executive kernels. To this date, executives kernels are provided for: TMS320C40, PIC18F2680, i80386, MC68332, MPC555, i80C196 and Unix/Linux workstations. Executive kernels for other processors can be achieved at reasonable cost following these examples as patterns.
FUNCTIONAL DESCRIPTION: Software for optimising the implementation of embedded distributed real-time applications and generating efficient and correct by construction code
NEWS OF THE YEAR: We improved the distribution and scheduling heuristics to take into account the needs of co-simulation.

- Participant: Yves Sorel
- Contact: Yves Sorel
- URL: http://www.syndex.org

6.2. EVT Kopernic

KEYWORDS: Embedded systems - Worst Case Execution Time - Real-time application - Statistics
SCIENTIFIC DESCRIPTION: The EVT-Kopernic tool is an implementation of the Extreme Value Theory (EVT) for the problem of the statistical estimation of worst-case bounds for the execution time of a program on a processor. Our implementation uses the two versions of EVT - GEV and GPD - to propose two independent methods of estimation. Their results are compared and only results that are sufficiently close allow to validate an estimation. Our tool is proved predictable by its unique choice of block (GEV) and threshold (GPD) while proposant reproducible estimations.

FUNCTIONAL DESCRIPTION: EVT-Kopernic is tool proposing a statistical estimation for bounds on worst-case execution time of a program on a processor. The estimator takes into account dependences between execution times by learning from the history of execution, while dealing also with cases of small variability of the execution times.
NEWS OF THE YEAR: Any statistical estimator should come with an representative measurement protocole based on the processus of composition, proved correct. We propose the first such principle of composition while using a Bayesien modeling taking into account iteratively different measurement models. The composition model has been described in a patent submitted this year with a scientific publication under preparation.

- Participants: Adriana Gogonel and Liliana Cucu
- Contact: Adriana Gogonel
- URL: http://inria-rscript.serveftp.com/

6.3. LoPhT-manycore

Logical to Physical Time compiler for many cores
KEYWORDS: Real time - Compilation - Task scheduling - Automatic parallelization
SCIENTIFIC DESCRIPTION: Lopht is a system-level compiler for embedded systems, whose objective is to fully automate the implementation process for certain classes of embedded systems. Like in a classical compiler (e.g. gcc), its input is formed of two objects. The first is a program providing a platform-indepedent description of the functionality to implement and of the non-functional requirements it must satisfy (e.g. real-time, partitioning). This is provided under the form of a data-flow synchronous program annotated with non-functional requirements. The second is a description of the implementation platform, defining the topology of the platform, the capacity of its elements, and possibly platform-dependent requirements (e.g. allocation).

From these inputs, Lopht produces all the C code and configuration information needed to allow compilation and execution on the physical target platform. Implementations are correct by construction Resulting implementations are functionally correct and satisfy the non-functional requirements. Lopht-manycore is a version of Lopht targeting shared-memory many-core architectures.

The algorithmic core of Lopht-manycore is formed of timing analysis, allocation, scheduling, and code generation heuristics which rely on four fundamental choices. 1) A static (off-line) real-time scheduling approach where allocation and scheduling are represented using time tables (also known as scheduling or reservation tables). 2) Scalability, attained through the use of low-complexity heuristics for all synthesis and associated analysis steps. 3) Efficiency (of generated implementations) is attained through the use of precise representations of both functionality and the platform, which allow for fine-grain allocation of resources such as CPU, memory, and communication devices such as network-on-chip multiplexers. 4) Full automation, including that of the timing analysis phase.

The last point is characteristic to Lopht-manycore. Existing methods for schedulability analysis and real-time software synthesis assume the existence of a high-level timing characterization that hides much of the hardware complexity. For instance, a common hypothesis is that synchronization and interference costs are accounted for in the duration of computations. However, the high-level timing characterization is seldom (if ever) soundly derived from the properties of the platform and the program. In practice, large margins (e.g. 100%) with little formal justification are added to computation durations to account for hidden hardware complexity. Lopht-manycore overcomes this limitation. Starting from the worst-case execution time (WCET) estimations of computation operations and from a precise and safe timing model of the platform, it maintains a precise timing accounting throughout the mapping process. To do this, timing accounting must take into account all details of allocation, scheduling, and code generation, which in turn must satisfy specific hypotheses.

FUNCTIONAL DESCRIPTION: Accepted input languages for functional specifications include dialects of Lustre such as Heptagon and Scade v4. To ensure the respect of real-time requirements, Lopht-manycore pilots the use of the worst-case execution time (WCET) analysis tool (ait from AbsInt). By doing this, and by using a precise timing model for the platform, Lopht-manycore eliminates the need to adjust the WCET values through the addition of margins to the WCET values that are usually both large and without formal safety guarantees. The output of Lopht-manycore is formed of all the multi-threaded C code and configuration information needed to allow compilation, linking/loading, and real-time execution on the target platform.
NEWS OF THE YEAR: In the framework of the ITEA3 ASSUME project we have extended the Lopht-manycore to allow multiple cores to access the same memory bank at the same time. To do this, the timing accounting of Lopht has been extended to take into account memory access interferences during the allocation and scheduling process. Lopht now also pilots the aiT static WCET analysis tool from AbsInt by generating the analysis scripts, thus ensuring the consistency between the hypotheses made by Lopht and the way timing analysis is performed by aiT. As a result, we are now able to synthesize code for the computing clusters of the Kalray MPPA256 platform. Lopht-manycore is evaluated on avionics case studies in the perspective of increasing its technology readiness level for this application class.

- Participants: Dumitru Potop-Butucaru and Keryan Didier
- Contact: Dumitru Potop-Butucaru

HYCOMES Project-Team

4. New Software and Platforms

4.1. Demodocos

Demodocos (Examples to Generic Scenario Models Generator)
KEYWORDS: Surgical process modelling - Net synthesis - Process mining
SCIENTIFIC DESCRIPTION: Demodocos is used to construct a Test and Flip net (Petri net variant) from a collection of instances of a given procedure. The tool takes as input either standard XES log files (a standard XML file format for process mining tools) or a specific XML file format for surgical applications. The result is a Test and Flip net and its marking graph. The tool can also build a #SEVEN scenario for integration into a virtual reality environment. The scenario obtained corresponds to the generalization of the input instances, namely the instances synthesis enriched with new behaviors respecting the relations of causality, conflicts and competition observed.

Demodocos is a synthesis tool implementing a linear algebraic polynomial time algorithm. Computations are done in the Z/2Z ring. Test and Flip nets extend Elementary Net Systems by allowing test to zero, test to one and flip arcs. The effect of flip arcs is to complement the marking of the place. While the net synthesis problem has been proved to be NP hard for Elementary Net Systems, thanks to flip arcs, the synthesis of Test and Flip nets can be done in polynomial time. Test and flip nets have the required expressivity to give concise and accurate representations of surgical processes (models of types of surgical operations). Test and Flip nets can express causality and conflict relations. The tool takes as input either standard XES log files (a standard XML file format for process mining tools) or a specific XML file format for surgical applications. The output is a Test and Flip net, solution of the following synthesis problem: Given a finite input language (log file), compute a net, which language is the least language in the class of Test and Flip net languages, containing the input language.
FUNCTIONAL DESCRIPTION: The tool Demodocos allows to build a generic model for a given procedure from some examples of instances of this procedure. The generated model can take the form of a graph, a Test 'n Flip net or a SEVEN scenario (intended for integration into a virtual reality environment).

The classic use of the tool is to apply the summary operation to a set of files describing instances of the target procedure. Several file formats are supported, including the standard XES format for log events. As output, several files are generated. These files represent the generic procedure in different forms, responding to varied uses.

This application is of limited interest in the case of an isolated use, out of context and without a specific objective when using the model generated. It was developed as part of a research project focusing in particular on surgical procedures, and requiring the generation of a generic model for integration into a virtual reality training environment. It is also quite possible to apply the same method in another context.

- Participants: Aurélien Lamercerie and Benoît Caillaud
- Contact: Benoît Caillaud
- Publication: Surgical Process Mining with Test and Flip Net Synthesis
- URL: http://tinyurl.com/oql6f3y

4.2. MICA

Model Interface Compositional Analysis Library
KEYWORDS: Modal interfaces - Contract-based desing

SCIENTIFIC DESCRIPTION: In Mica, systems and interfaces are represented by extension. However, a careful design of the state and event heap enables the definition, composition and analysis of reasonably large systems and interfaces. The heap stores states and events in a hash table and ensures structural equality (there is no duplication). Therefore complex data-structures for states and events induce a very low overhead, as checking equality is done in constant time.

Thanks to the Inter module and the mica interactive environment, users can define complex systems and interfaces using Ocaml syntax. It is even possible to define parameterized components as Ocaml functions.
FUNCTIONAL DESCRIPTION: Mica is an Ocaml library implementing the Modal Interface algebra. The purpose of Modal Interfaces is to provide a formal support to contract based design methods in the field of system engineering. Modal Interfaces enable compositional reasoning methods on I/O reactive systems.

- Participant: Benoît Caillaud
- Contact: Benoît Caillaud
- URL: http://www.irisa.fr/s4/tools/mica/

4.3. TnF-C++

FUNCTIONAL DESCRIPTION: TnF-C++ is a robust and portable re-implementation of Flipflop, developed in 2014 and integrated in the S3PM toolchain. Both software have been designed in the context of the S3PM project on surgical procedure modeling and simulation,

- Contact: Benoît Caillaud

KAIROS Team

5. New Software and Platforms

5.1. Vercors Component Editor (VCE)

VERification of models for distributed communicating COmponants, with safety and Security
FUNCTIONAL DESCRIPTION: The Vercors tools include front-ends for specifying the architecture and behaviour of components in the form of UML diagrams. We translate these high-level specifications, into behavioural models in various formats, and we also transform these models using abstractions. In a final step, abstract models are translated into the input format for various verification toolsets. Currently we mainly use the various analysis modules of the CADP toolset.

RELEASE FUNCTIONAL DESCRIPTION: It includes integrated graphical editors for GCM component architecture descriptions, UML classes, interfaces, and state-machines. The user diagrams can be checked using the recently published validation rules from, then the corresponding GCM components can be executed using an automatic generation of the application ADL, and skeletons of Java files.

- Participants: Antonio Cansado, Bartlomiej Szejna, Eric Madelaine, Ludovic Henrio, Marcela Rivera, Nassim Jibai, Oleksandra Kulankhina and Siqi Li
- Contact: Eric Madelaine
- URL: http://www-sop.inria.fr/oasis/index.php?page=vercors

5.2. TimeSquare

KEYWORDS: Profil MARTE - Embedded systems - UML - IDM
SCIENTIFIC DESCRIPTION: TimeSquare offers six main functionalities:

* graphical and/or textual interactive specification of logical clocks and relative constraints between them,

* definition and handling of user-defined clock constraint libraries,

* automated simulation of concurrent behavior traces respecting such constraints, using a Boolean solver for consistent trace extraction,

* call-back mechanisms for the traceability of results (animation of models, display and interaction with waveform representations, generation of sequence diagrams...).

* compilation to pure java code to enable embedding in non eclipse applications or to be integrated as a time and concurrency solver within an existing tool.

* a generation of the whole state space of a specification (if finite of course) in order to enable model checking of temporal properties on it
FUNCTIONAL DESCRIPTION: TimeSquare is a software environment for the modeling and analysis of timing constraints in embedded systems. It relies specifically on the Time Model of the Marte UML profile, and more accurately on the associated Clock Constraint Specification Language (CCSL) for the expression of timing constraints.

- Participants: Benoît Ferrero, Charles André, Frédéric Mallet, Julien Deantoni and Nicolas Chleq
- Contact: Julien Deantoni
- URL: http://timesquare.inria.fr

5.3. GEMOC Studio

KEYWORDS: DSL - Language workbench - Model debugging

SCIENTIFIC DESCRIPTION: The language workbench put together the following tools seamlessly integrated to the Eclipse Modeling Framework (EMF):

- Melange, a tool-supported meta-language to modularly define executable modeling languages with execution functions and data, and to extend (EMF-based) existing modeling languages. - MoCCML, a tool-supported meta-language dedicated to the specification of a Model of Concurrency and Communication (MoCC) and its mapping to a specific abstract syntax and associated execution functions of a modeling language. - GEL, a tool-supported meta-language dedicated to the specification of the protocol between the execution functions and the MoCC to support the feedback of the data as well as the callback of other expected execution functions. - BCOoL, a tool-supported meta-language dedicated to the specification of language coordination patterns to automatically coordinates the execution of, possibly heterogeneous, models. - Sirius Animator, an extension to the model editor designer Sirius to create graphical animators for executable modeling languages.
FUNCTIONAL DESCRIPTION: The GEMOC Studio is an eclipse package that contains components supporting the GEMOC methodology for building and composing executable Domain-Specific Modeling Languages (DSMLs). It includes the two workbenches: The GEMOC Language Workbench: intended to be used by language designers (aka domain experts), it allows to build and compose new executable DSMLs. The GEMOC Modeling Workbench: intended to be used by domain designersto create, execute and coordinate models conforming to executable DSMLs. The different concerns of a DSML, as defined with the tools of the language workbench, are automatically deployed into the modeling workbench. They parametrize a generic execution framework that provide various generic services such as graphical animation, debugging tools, trace and event managers, timeline, etc.

- Participants: Didier Vojtisek, Dorian Leroy, Erwan Bousse, Fabien Coulon and Julien Deantoni
- Partners: IRIT - ENSTA - I3S - OBEO - Thales TRT
- Contact: Benoît Combemale
- URL: http://gemoc.org/studio.html

5.4. BCOol

BCOoL
KEYWORDS: DSL - Language workbench - Behavior modeling - Model debugging - Model animation
FUNCTIONAL DESCRIPTION: BCOoL is a tool-supported meta-language dedicated to the specification of language coordination patterns to automatically coordinates the execution of, possibly heterogeneous, models.

- Participants: Julien Deantoni, Matias Vara Larsen, Benoît Combemale and Didier Vojtisek
- Contact: Julien Deantoni
- URL: http://www.gemoc.org

5.5. MoCCML

KEYWORDS: DSL - Language workbench - Modeling workbench - Model debugging - Model animation
FUNCTIONAL DESCRIPTION: The MoCCML / Concurrency provides components and engines supporting concurrency and/or time in execution semantics.

- Participants: Julien Deantoni, Didier Vojtisek, Joël Champeau, Benoît Combemale and Stephen Creff
- Partner: ENSTA
- Contact: Benoît Combemale
- URL: http://www.gemoc.org

PARKAS Project-Team

5. New Software and Platforms

5.1. Cmmtest

FUNCTIONAL DESCRIPTION: Cmmtest is a tool for hunting concurrency compiler bugs. The Cmmtest tool performs random testing of C and C++ compilers against the C11/C++11 memory model. A test case is any well-defined, sequential C program, for each test case, cmmtest:

compiles the program using the compiler and compiler optimisations that are being tested,

runs the compiled program in an instrumented execution environment that logs all memory accesses to global variables and synchronisations,

compares the recorded trace with a reference trace for the same program, checking if the recorded trace can be obtained from the reference trace by valid eliminations, reorderings and introductions.

Cmmtest identified several mistaken write introductions and other unexpected behaviours in the latest release of the gcc compiler. These have been promptly fixed by the gcc developers.

- Participants: Anirudh Kumar, Francesco Zappa Nardelli, Pankaj More, Pankaj Pawan, Pankaj Prateek Kewalramani and Robin Morisset
- Contact: Francesco Zappa Nardelli
- URL: http://www.di.ens.fr/~zappa/projects/cmmtest/

5.2. GCC

KEYWORDS: Compilation - Polyhedral compilation
FUNCTIONAL DESCRIPTION: The GNU Compiler Collection includes front ends for C, C++, Objective-C, Fortran, Java, Ada, and Go, as well as libraries for these languages (libstdc++, libgcj,...). GCC was originally written as the compiler for the GNU operating system. The GNU system was developed to be 100

- Participants: Albert Cohen, Feng Li, Nhat Minh Le, Riyadh Baghdadi and Tobias Grosser
- Contact: Albert Cohen
- URL: http://gcc.gnu.org/

5.3. Heptagon

KEYWORDS: Compilers - Synchronous Language - Controller synthesis
FUNCTIONAL DESCRIPTION: Heptagon is an experimental language for the implementation of embedded real-time reactive systems. It is developed inside the Synchronics large-scale initiative, in collaboration with Inria Rhones-Alpes. It is essentially a subset of Lucid Synchrone, without type inference, type polymorphism and higher-order. It is thus a Lustre-like language extended with hierchical automata in a form very close to SCADE 6. The intention for making this new language and compiler is to develop new aggressive optimization techniques for sequential C code and compilation methods for generating parallel code for different platforms. This explains much of the simplifications we have made in order to ease the development of compilation techniques.

The current version of the compiler includes the following features: - Inclusion of discrete controller synthesis within the compilation: the language is equipped with a behavioral contract mechanisms, where assumptions can be described, as well as an "enforce" property part. The semantics of this latter is that the property should be enforced by controlling the behaviour of the node equipped with the contract. This property will be enforced by an automatically built controller, which will act on free controllable variables given by the programmer. This extension has been named BZR in previous works. - Expression and compilation of array values with modular memory optimization. The language allows the expression and operations on arrays (access, modification, iterators). With the use of location annotations, the programmer can avoid unnecessary array copies.

- Participants: Adrien Guatto, Brice Gelineau, Cédric Pasteur, Eric Rutten, Gwenaël Delaval, Léonard Gérard and Marc Pouzet
- Partners: UGA - ENS Paris - Inria - LIG
- Contact: Gwenaël Delaval
- URL: http://heptagon.gforge.inria.fr

5.4. isl

FUNCTIONAL DESCRIPTION: isl is a library for manipulating sets and relations of integer points bounded by linear constraints. Supported operations on sets include intersection, union, set difference, emptiness check, convex hull, (integer) affine hull, integer projection, transitive closure (and over-approximation), computing the lexicographic minimum using parametric integer programming. It includes an ILP solver based on generalized basis reduction, and a new polyhedral code generator. isl also supports affine transformations for polyhedral compilation, and increasingly abstract representations to model source and intermediate code in a polyhedral framework.

- Participants: Albert Cohen, Sven Verdoolaege and Tobias Grosser
- Contact: Sven Verdoolaege
- URL: http://freshmeat.net/projects/isl

5.5. Lem

lightweight executable mathematics
FUNCTIONAL DESCRIPTION: Lem is a lightweight tool for writing, managing, and publishing large scale semantic definitions. It is also intended as an intermediate language for generating definitions from domain-specific tools, and for porting definitions between interactive theorem proving systems (such as Coq, HOL4, and Isabelle). As such it is a complementary tool to Ott. Lem resembles a pure subset of Objective Caml, supporting typical functional programming constructs, including top-level parametric polymorphism, datatypes, records, higher-order functions, and pattern matching. It also supports common logical mechanisms including list and set comprehensions, universal and existential quantifiers, and inductively defined relations. From this, Lem generates OCaml, HOL4, Coq, and Isabelle code.

- Participants: Francesco Zappa Nardelli, Peter Sewell and Scott Owens
- Contact: Francesco Zappa Nardelli
- URL: http://www.cl.cam.ac.uk/~pes20/lem/

5.6. Lucid Synchrone

FUNCTIONAL DESCRIPTION: Lucid Synchrone is a language for the implementation of reactive systems. It is based on the synchronous model of time as provided by Lustre combined with features from ML languages. It provides powerful extensions such as type and clock inference, type-based causality and initialization analysis and allows to arbitrarily mix data-flow systems and hierarchical automata or flows and valued signals.

RELEASE FUNCTIONAL DESCRIPTION: The language is still used for teaching and in our research but we do not develop it anymore. Nonetheless, we have integrated several features from Lucid Synchrone in new research prototypes described below. The Heptagon language and compiler are a direct descendent of it. The new language Zélus for hybrid systems modeling borrows many features originaly introduced in Lucid Synchrone.

- Contact: Marc Pouzet
- URL: http://www.di.ens.fr/~pouzet/lucid-synchrone/

5.7. Lucy-n

Lucy-n: an n-synchronous data-flow programming language
FUNCTIONAL DESCRIPTION: Lucy-n is a language to program in the n-synchronous model. The language is similar to Lustre with a buffer construct. The Lucy-n compiler ensures that programs can be executed in bounded memory and automatically computes buffer sizes. Hence this language allows to program Kahn networks, the compiler being able to statically compute bounds for all FIFOs in the program.

- Participants: Adrien Guatto, Albert Cohen, Louis Mandel and Marc Pouzet
- Contact: Albert Cohen
- URL: https://www.lri.fr/~mandel/lucy-n/

5.8. Ott

FUNCTIONAL DESCRIPTION: Ott is a tool for writing definitions of programming languages and calculi. It takes as input a definition of a language syntax and semantics, in a concise and readable ASCII notation that is close to what one would write in informal mathematics. It generates output:

a LaTeX source file that defines commands to build a typeset version of the definition,

a Coq version of the definition,

an Isabelle version of the definition, and

a HOL version of the definition.

Additionally, it can be run as a filter, taking a LaTeX/Coq/Isabelle/HOL source file with embedded (symbolic) terms of the defined language, parsing them and replacing them by typeset terms.

The main goal of the Ott tool is to support work on large programming language definitions, where the scale makes it hard to keep a definition internally consistent, and to keep a tight correspondence between a definition and implementations. We also wish to ease rapid prototyping work with smaller calculi, and to make it easier to exchange definitions and definition fragments between groups. The theorem-prover backends should enable a smooth transition between use of informal and formal mathematics.

- Participants: Francesco Zappa Nardelli, Peter Sewell and Scott Owens
- Contact: Francesco Zappa Nardelli
- URL: http://www.cl.cam.ac.uk/~pes20/ott/

5.9. PPCG

FUNCTIONAL DESCRIPTION: PPCG is our source-to-source research tool for automatic parallelization in the polyhedral model. It serves as a test bed for many compilation algorithms and heuristics published by our group, and is currently the best automatic parallelizer for CUDA and OpenCL (on the Polybench suite).

- Participants: Albert Cohen, Riyadh Baghdadi, Sven Verdoolaege and Tobias Grosser
- Contact: Sven Verdoolaege
- URL: http://freshmeat.net/projects/ppcg

5.10. ReactiveML

FUNCTIONAL DESCRIPTION: ReactiveML is a programming language dedicated to the implementation of interactive systems as found in graphical user interfaces, video games or simulation problems. ReactiveML is based on the synchronous reactive model due to Boussinot, embedded in an ML language (OCaml).

The Synchronous reactive model provides synchronous parallel composition and dynamic features like the dynamic creation of processes. In ReactiveML, the reactive model is integrated at the language level (not as a library) which leads to a safer and a more natural programming paradigm.

- Participants: Cédric Pasteur, Guillaume Baudart and Louis Mandel
- Contact: Guillaume Baudart

5.11. SundialsML

Sundials/ML
KEYWORDS: Simulation - Mathematics - Numerical simulations
SCIENTIFIC DESCRIPTION: Sundials/ML is a comprehensive OCaml interface to the Sundials suite of numerical solvers (CVODE, CVODES, IDA, IDAS, KINSOL). Its structure mostly follows that of the Sundials library, both for ease of reading the existing documentation and for adapting existing source code, but several changes have been made for programming convenience and to increase safety, namely:

solver sessions are mostly configured via algebraic data types rather than multiple function calls,

errors are signalled by exceptions not return codes (also from user-supplied callback routines),

user data is shared between callback routines via closures (partial applications of functions),

vectors are checked for compatibility (using a combination of static and dynamic checks), and

explicit free commands are not necessary since OCaml is a garbage-collected language.
FUNCTIONAL DESCRIPTION: Sundials/ML is a comprehensive OCaml interface to the Sundials suite of numerical solvers (CVODE, CVODES, IDA, IDAS, KINSOL, ARKODE).

- Participants: Jun Inoue, Marc Pouzet and Timothy Bourke
- Partner: UPMC
- Contact: Marc Pouzet
- URL: http://inria-parkas.github.io/sundialsml/

5.12. Zelus

SCIENTIFIC DESCRIPTION: The Zélus implementation has two main parts: a compiler that transforms Zélus programs into OCaml programs and a runtime library that orchestrates compiled programs and numeric solvers. The runtime can use the Sundials numeric solver, or custom implementations of well-known algorithms for numerically approximating continuous dynamics.
FUNCTIONAL DESCRIPTION: Zélus is a new programming language for hybrid system modeling. It is based on a synchronous language but extends it with Ordinary Differential Equations (ODEs) to model continuous-time behaviors. It allows for combining arbitrarily data-flow equations, hierarchical automata and ODEs. The language keeps all the fundamental features of synchronous languages: the compiler statically ensure the absence of deadlocks and critical races, it is able to generate statically scheduled code running in bounded time and space and a type-system is used to distinguish discrete and logical-time signals from continuous-time ones. The ability to combines those features with ODEs made the language usable both for programming discrete controllers and their physical environment.

- Participants: Marc Pouzet and Timothy Bourke
- Contact: Marc Pouzet

SPADES Project-Team

5. New Software and Platforms

5.1. pyCPA_TCA

FUNCTIONAL DESCRIPTION: We are developing pyCPA_TCA , a pyCPA plugin for Typical Worst-Case Analysis as described in Section. pyCPA is an open-source Python implementation of Compositional Performance Analysis developed at TU Braunschweig, which allows in particular response-time analysis. pyCPA_TCA is an extension of this tool that is co-developed by Sophie Quinton and Zain Hammadeh at TU Braunschweig. It allows in particular the computation of weakly-hard guarantees for real-time tasks, i.e. number of deadline misses out of a sequence of executions. So far, pyCPA_TCA is restricted to uniprocessor systems of independent tasks, scheduled according to static priority scheduling.

- Contact: Sophie Quinton

5.2. LDDL

Coq proofs of circuit transformations for fault-tolerance
KEYWORDS: Fault-tolerance - Transformation - Coq - Semantics
FUNCTIONAL DESCRIPTION: We have been developing a Coq-based framework to formally verify the functional and fault-tolerance properties of circuit transformations. Circuits are described at the gate level using LDDL, a Low-level Dependent Description Language inspired from muFP. Our combinator language, equipped with dependent types, ensures that circuits are well-formed by construction (gates correctly plugged, no dangling wires, no combinational loops, . . .). Faults like Single-Event Upsets (SEUs) (i.e., bit-flips in flipflops) and SETs (i.e., glitches propagating in the combinational circuit) and fault-models like "at most 1 SEU or SET within n clock cycles" are described in the operational semantics of LDDL. Fault-tolerance techniques are described as transformations of LDDL circuits.

The framework has been used to prove the correctness of three fault-tolerance techniques: TMR, TTR and DTR. The size of specifications and proofs for the common part (LDDL syntax and semantics, libraries) is 5000 lines of Coq (excluding comments and blank lines), 700 for TMR, 3500 for TTR and 7000 for DTR.

- Authors: Pascal Fradet and Dmitry Burlyaev
- Contact: Pascal Fradet
- URL: https://team.inria.fr/spades/fthwproofs/

TEA Project-Team

6. New Software and Platforms

6.1. ADFG

Affine data-flow graphs schedule synthesizer
KEYWORDS: Code generation - Scheduling - Static program analysis
FUNCTIONAL DESCRIPTION: ADFG is a synthesis tool of real-time system scheduling parameters: ADFG computes task periods and buffer sizes of systems resulting in a trade-off between throughput maximization and buffer size minimization. ADFG synthesizes systems modeled by ultimately cyclo-static dataflow (UCSDF) graphs, an extension of the standard CSDF model.

Knowing the WCET (Worst Case Execute Time) of the actors and their exchanges on the channels, ADFG tries to synthezise the scheduler of the application. ADFG offers several scheduling policies and can detect unschedulable systems. It ensures that the real scheduling does not cause overflows or underflows and tries to maximize the throughput (the processors utilization) while minimizing the storage space needed between the actors (i.e. the buffer sizes).

Abstract affine scheduling is first applied on the dataflow graph, that consists only of periodic actors, to compute timeless scheduling constraints (e.g. relation between the speeds of two actors) and buffering parameters. Then, symbolic schedulability policies analysis (i.e., synthesis of timing and scheduling parameters of actors) is applied to produce the scheduller for the actors.

ADFG, initially defined to synthesize real-time schedulers for SCJ/L1 applications, may be used for scheduling analysis of AADL programs.

- Authors: Thierry Gautier, Jean-Pierre Talpin, Adnan Bouakaz, Alexandre Honorat and Loïc Besnard
- Contact: Loïc Besnard

6.2. POLYCHRONY

KEYWORDS: Code generation - AADL - Proof - Optimization - Multi-clock - GALS - Architecture - Cosimulation - Real time - Synchronous Language
FUNCTIONAL DESCRIPTION: Polychrony is an Open Source development environment for critical/embedded systems. It is based on Signal, a real-time polychronous data-flow language. It provides a unified model-driven environment to perform design exploration by using top-down and bottom-up design methodologies formally supported by design model transformations from specification to implementation and from synchrony to asynchrony. It can be included in heterogeneous design systems with various input formalisms and output languages. The Polychrony tool-set provides a formal framework to: validate a design at different levels, by the way of formal verification and/or simulation, refine descriptions in a top-down approach, abstract properties needed for black-box composition, compose heterogeneous components (bottom-up with COTS), generate executable code for various architectures. The Polychrony tool-set contains three main components and an experimental interface to GNU Compiler Collection (GCC):

* The Signal toolbox, a batch compiler for the Signal language, and a structured API that provides a set of program transformations. Itcan be installed without other components and is distributed under GPL V2 license.

* The Signal GUI, a Graphical User Interface to the Signal toolbox (editor + interactive access to compiling functionalities). It can be used either as a specific tool or as a graphical view under Eclipse. It has been transformed and restructured, in order to get a more up-to-date interface allowing multi-window manipulation of programs. It is distributed under GPL V2 license.

* The POP Eclipse platform, a front-end to the Signal toolbox in the Eclipse environment. It is distributed under EPL license.

- Participants: Loïc Besnard, Paul Le Guernic and Thierry Gautier
- Partners: CNRS - Inria
- Contact: Loïc Besnard
- URL: https://www.polarsys.org/projects/polarsys.pop

6.3. Polychrony AADL2SIGNAL

KEYWORDS: Real-time application - Polychrone - Synchronous model - Polarsys - Polychrony - Signal - AADL - Eclipse - Meta model
FUNCTIONAL DESCRIPTION: This polychronous MoC has been used previously as semantic model for systems described in the core AADL standard. The core AADL is extended with annexes, such as the Behavior Annex, which allows to specify more precisely architectural behaviors. The translation from AADL specifications into the polychronous model should take into account these behavior specifications, which are based on description of automata.

For that purpose, the AADL state transition systems are translated as Signal automata (a slight extension of the Signal language has been defined to support the model of polychronous automata).

Once the AADL model of a system transformed into a Signal program, one can analyze the program using the Polychrony framework in order to check if timing, scheduling and logical requirements over the whole system are met.

We have implemented the translation and experimented it using a concrete case study, which is the AADL modeling of an Adaptive Cruise Control (ACC) system, a highly safety-critical system embedded in recent cars.

- Participants: Huafeng Yu, Loïc Besnard, Paul Le Guernic, Thierry Gautier and Yue Ma
- Partner: CNRS
- Contact: Loïc Besnard
- URL: http://www.inria.fr/equipes/tea

6.4. POP

Polychrony on Polarsys
KEYWORDS: Synchronous model - Model-driven engineering
FUNCTIONAL DESCRIPTION: The Eclipse project POP is a model-driven engineering front-end to our open-source toolset Polychrony. a major achievement of the ESPRESSO (and now TEA) project-team. The Eclipse project POP is a model-driven engineering front-end to our open-source toolset Polychrony. It was finalised in the frame of project OPEES, as a case study: by passing the POLARSYS qualification kit as a computer aided simulation and verification tool. This qualification was implemented by CS Toulouse in conformance with relevant generic (platform independent) qualification documents. Polychrony is now distributed by the Eclipse project POP on the platform of the POLARSYS industrial working group. Team TEA aims at continuing its dissemination to academic partners, as to its principles and features, and industrial partners, as to the services it can offer.

Project POP is composed of the Polychrony tool set, under GPL license, and its Eclipse framework, under EPL license. SSME (Syntactic Signal-Meta under Eclipse), is the meta-model of the Signal language implemented with Eclipse/Ecore. It describes all syntactic elements specified in Signal Reference Manual [0]: all Signal operators (e.g. arithmetic, clock synchronization), model (e.g. process frame, module), and construction (e.g. iteration, type declaration). The meta-model primarily aims at making the language and services of the Polychrony environment available to inter-operation and composition with other components (e.g. AADL, Simulink, GeneAuto, P) within an Eclipse-based development tool-chain. Polychrony now comprises the capability to directly import and export Ecore models instead of textual Signal programs, in order to facilitate interaction between components within such a tool-chain. The download site for project POP has opened in 2015 at https://www.polarsys.org/projects/polarsys.pop. It should be noted that the Eclipse Foundation does not host code under GPL license. So, the Signal toolbox useful to compile Signal code from Eclipse is hosted on our web server.

- Participants: Jean-Pierre Talpin, Loïc Besnard, Paul Le Guernic and Thierry Gautier
- Contact: Loïc Besnard
- URL: https://www.polarsys.org/projects/polarsys.pop

6.5. Sigali

FUNCTIONAL DESCRIPTION: Sigali is a model-checking tool that operates on ILTS (Implicit Labeled Transition Systems, an equational representation of an automaton), an intermediate model for discrete event systems. It offers functionalities for verification of reactive systems and discrete controller synthesis. The techniques used consist in manipulating the system of equations instead of the set of solutions, which avoids the enumeration of the state space. Each set of states is uniquely characterized by a predicate and the operations on sets can be equivalently performed on the associated predicates. Therefore, a wide spectrum of properties, such as liveness, invariance, reachability and attractivity, can be checked. Algorithms for the computation of predicates on states are also available. Sigali is connected with the Polychrony environment (Tea project-team) as well as the Matou environment (VERIMAG), thus allowing the modeling of reactive systems by means of Signal Specification or Mode Automata and the visualization of the synthesized controller by an interactive simulation of the controlled system.

- Contact: Hervé Marchand

0

SIGNAL V4-Inria version: Reference Manual. Besnard, L., Gautier, T. and Le Guernic, P. http://www.irisa.fr/espresso/Polychrony, 2010

ANTIQUE Project-Team

6. New Software and Platforms

6.1. APRON

SCIENTIFIC DESCRIPTION: The APRON library is intended to be a common interface to various underlying libraries/abstract domains and to provide additional services that can be implemented independently from the underlying library/abstract domain, as shown by the poster on the right (presented at the SAS 2007 conference. You may also look at:
FUNCTIONAL DESCRIPTION: 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.

- Participants: Antoine Miné and Bertrand Jeannet
- Contact: Antoine Miné
- URL: http://apron.cri.ensmp.fr/library/

6.2. Astrée

The AstréeA Static Analyzer of Asynchronous Software
KEYWORDS: Static analysis - Static program analysis - Program verification - Software Verification - Abstraction
SCIENTIFIC DESCRIPTION: Astrée analyzes structured C programs, with complex memory usages, but without dynamic memory allocation nor 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),

failure of user-defined assertions.
FUNCTIONAL DESCRIPTION: Astrée analyzes structured C programs, with complex memory usages, but without dynamic memory allocation nor 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), - failure of user-defined assertions.

Astrée is a static analyzer for sequential programs based on abstract interpretation. The Astrée static analyzer aims at proving the absence of runtime errors in programs written in the C programming language.

- Participants: Antoine Miné, Jérôme Feret, Laurent Mauborgne, Patrick Cousot, Radhia Cousot and Xavier Rival
- Partners: CNRS - ENS Paris - AbsInt Angewandte Informatik GmbH
- Contact: Patrick Cousot
- URL: http://www.astree.ens.fr/

6.3. AstréeA

The AstréeA Static Analyzer of Asynchronous Software
KEYWORDS: Static analysis - Static program analysis
SCIENTIFIC DESCRIPTION: AstréeA 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. AstréeA 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, AstréeA 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). AstréeA checks for the same run-time errors as Astrée , with the addition of data-races.
FUNCTIONAL DESCRIPTION: AstréeA is a static analyzer prototype for parallel software based on abstract interpretation. The AstréeA prototype is a fork of the Astrée static analyzer that adds support for analyzing parallel embedded C software.

- Participants: Antoine Miné, Jérôme Feret, Patrick Cousot, Radhia Cousot and Xavier Rival
- Partners: CNRS - ENS Paris - AbsInt Angewandte Informatik GmbH
- Contact: Patrick Cousot
- URL: http://www.astreea.ens.fr/

6.4. ClangML

KEYWORD: Compilation
FUNCTIONAL DESCRIPTION: ClangML is an OCaml binding with the Clang front-end of the LLVM compiler suite. Its goal is to provide an easy to use solution to parse a wide range of C programs, that can be called from static analysis tools implemented in OCaml, which allows to test them on existing programs written in C (or in other idioms derived from C) without having to redesign a front-end from scratch. ClangML features an interface to a large set of internal AST nodes of Clang , with an easy to use API. Currently, ClangML supports all C language AST nodes, as well as a large part of the C nodes related to C++ and Objective-C.

- Participants: Devin Mccoughlin, François Berenger and Pippijn Van Steenhoven
- Contact: Xavier Rival
- URL: https://github.com/Antique-team/clangml/tree/master/clang

6.5. FuncTion

SCIENTIFIC DESCRIPTION: FuncTion is based on an extension to liveness properties of the framework to analyze termination by abstract interpretation proposed by Patrick Cousot and Radhia Cousot. FuncTion infers ranking functions using piecewise-defined abstract domains. Several domains are available to partition the ranking function, including intervals, octagons, and polyhedra. Two domains are also available to represent the value of ranking functions: a domain of affine ranking functions, and a domain of ordinal-valued ranking functions (which allows handling programs with unbounded non-determinism).

FUNCTIONAL DESCRIPTION: FuncTion is a research prototype static analyzer to analyze the termination and functional liveness properties of programs. It accepts programs in a small non-deterministic imperative language. It is also parameterized by a property: either termination, or a recurrence or a guarantee property (according to the classification by Manna and Pnueli of program properties). It then performs a backward static analysis that automatically infers sufficient conditions at the beginning of the program so that all executions satisfying the conditions also satisfy the property.

- Participants: Antoine Miné and Caterina Urban
- Contact: Caterina Urban
- URL: http://www.di.ens.fr/~urban/FuncTion.html

6.6. HOO

Heap Abstraction for Open Objects
FUNCTIONAL DESCRIPTION: JSAna with HOO is a static analyzer for JavaScript programs. The primary component, HOO, which is designed to be reusable by itself, is an abstract domain for a dynamic language heap. A dynamic language heap consists of open, extensible objects linked together by pointers. Uniquely, HOO abstracts these extensible objects, where attribute/field names of objects may be unknown. Additionally, it contains features to keeping precise track of attribute name/value relationships as well as calling unknown functions through desynchronized separation.

As a library, HOO is useful for any dynamic language static analysis. It is designed to allow abstractions for values to be easily swapped out for different abstractions, allowing it to be used for a wide-range of dynamic languages outside of JavaScript.

- Participant: Arlen Cox
- Contact: Arlen Cox

6.7. MemCAD

The MemCAD static analyzer
KEYWORDS: Static analysis - Abstraction
FUNCTIONAL DESCRIPTION: MemCAD is a static analyzer that focuses on memory abstraction. It takes as input C programs, and computes invariants on the data structures manipulated by the programs. It can also verify memory safety. It comprises several memory abstract domains, including a flat representation, and two graph abstractions with summaries based on inductive definitions of data-structures, such as lists and trees and several combination operators for memory abstract domains (hierarchical abstraction, reduced product). The purpose of this construction is to offer a great flexibility in the memory abstraction, so as to either make very efficient static analyses of relatively simple programs, or still quite efficient static analyses of very involved pieces of code. The implementation consists of over 30 000 lines of ML code, and relies on the ClangML front-end. The current implementation comes with over 300 small size test cases that are used as regression tests.

- Participants: Antoine Toubhans, François Berenger, Huisong Li and Xavier Rival
- Contact: Xavier Rival
- URL: http://www.di.ens.fr/~rival/memcad.html

6.8. OPENKAPPA

La platte-forme de modélisation OpenKappa
KEYWORDS: Model reduction - Simulation - Static analysis - Modeling - Systems Biology

SCIENTIFIC DESCRIPTION: OpenKappa is a collection of tools to build, debug and run models of biological pathways. It contains a compiler for the Kappa Language, a static analyzer (for debugging models), a simulator, a compression tool for causal traces, and a model reduction tool.

- Participants: Jean Krivine, Jérôme Feret, Kim Quyen Ly, Pierre Boutillier, Russ Harmer, Vincent Danos and Walter Fontana
- Partners: ENS Lyon - Université Paris-Diderot - HARVARD Medical School
- Contact: Jérôme Feret
- URL: http://www.kappalanguage.org/

6.9. QUICr

FUNCTIONAL DESCRIPTION: QUICr is an OCaml library that implements a parametric abstract domain for sets. It is constructed as a functor that accepts any numeric abstract domain that can be adapted to the interface and produces an abstract domain for sets of numbers combined with numbers. It is relational, flexible, and tunable. It serves as a basis for future exploration of set abstraction.

- Participant: Arlen Cox
- Contact: Arlen Cox

6.10. LCertify

KEYWORD: Compilation
SCIENTIFIC DESCRIPTION: 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.
FUNCTIONAL DESCRIPTION: Abstract interpretation, Certified compilation, Static analysis, Translation validation, Verifier. 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 semantic equivalence. 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.ntees that no compiler bug did cause incorrect code to be generated.

- Participant: Xavier Rival
- Partners: CNRS - ENS Paris
- Contact: Xavier Rival
- URL: http://www.di.ens.fr/~rival/lcertify.html

6.11. Zarith

FUNCTIONAL DESCRIPTION: Zarith is a small (10K lines) OCaml library that implements arithmetic and logical operations over arbitrary-precision integers. It is based on the GNU MP library to efficiently implement arithmetic over big integers. Special care has been taken to ensure the efficiency of the library also for small integers: small integers are represented as Caml unboxed integers and use a specific C code path. Moreover, optimized assembly versions of small integer operations are provided for a few common architectures.

Zarith is currently used in the Astrée analyzer to enable the sound analysis of programs featuring 64-bit (or larger) integers. It is also used in the Frama-C analyzer platform developed at CEA LIST and Inria Saclay.

- Participants: Antoine Miné, Pascal Cuoq and Xavier Leroy
- Contact: Antoine Miné
- URL: http://forge.ocamlcore.org/projects/zarith

CELTIQUE Project-Team

3. New Software and Platforms

3.1. Jacal

JAvaCard AnaLyseur
KEYWORDS: JavaCard - Certification - Static program analysis - AFSCM
FUNCTIONAL DESCRIPTION: Jacal is a JAvaCard AnaLyseur developed on top of the SAWJA platform. This proprietary software verifies automatically that Javacard programs conform with the security guidelines issued by the AFSCM (Association Française du Sans Contact Mobile). Jacal is based on the theory of abstract interpretation and combines several object-oriented and numeric analyses to automatically infer sophisticated invariants about the program behaviour. The result of the analysis is thereafter harvest to check that it is sufficient to ensure the desired security properties.

- Participants: David Pichardie, Delphine Demange, Frédéric Besson and Thomas Jensen
- Contact: Thomas Jensen

3.2. Javalib

FUNCTIONAL DESCRIPTION: Javalib is an efficient library to parse Java .class files into OCaml data structures, thus enabling the OCaml programmer to extract information from class files, to manipulate and to generate valid .class files.

- Participants: David Pichardie, Frédéric Besson, Laurent Guillo, Laurent Hubert, Nicolas Barré, Pierre Vittet and Tiphaine Turpin
- Contact: Frédéric Besson
- URL: http://sawja.inria.fr/

3.3. JSCert

Certified JavaScript
FUNCTIONAL DESCRIPTION: The JSCert project aims to really understand JavaScript. JSCert itself is a mechanised specification of JavaScript, written in the Coq proof assistant, which closely follows the ECMAScript 5 English standard. JSRef is a reference interpreter for JavaScript in OCaml , which has been proved correct with respect to JSCert and tested with the Test 262 test suite.

- Participants: Alan Schmitt and Martin Bodin
- Partner: Imperial College London
- Contact: Alan Schmitt
- URL: http://jscert.org/

3.4. SAWJA

Static Analysis Workshop for Java
KEYWORDS: Security - Software - Code review - Smart card
SCIENTIFIC DESCRIPTION: Sawja is a library written in OCaml, relying on Javalib to provide a high level representation of Java bytecode programs. It name comes from Static Analysis Workshop for JAva. Whereas Javalib is dedicated to isolated classes, Sawja handles bytecode programs with their class hierarchy and with control flow algorithms.

Moreover, Sawja provides some stackless intermediate representations of code, called JBir and A3Bir. The transformation algorithm, common to these representations, has been formalized and proved to be semantics-preserving.

See also the web page http://sawja.inria.fr/ .

Version: 1.5

Programming language: Ocaml
FUNCTIONAL DESCRIPTION: Sawja is a toolbox for developing static analysis of Java code in bytecode format. Sawja provides advanced algorithms for reconstructing high-level programme representations. The SawjaCard tool dedicated to JavaCard is based on the Sawja infrastructure and automatically validates the security guidelines issued by AFSCM (http://www.afscm.org/). SawjaCard can automate the code audit process and automatic verification of functional properties.

- Participants: David Pichardie, Frédéric Besson and Laurent Guillo
- Partners: CNRS - ENS Cachan
- Contact: Frédéric Besson
- URL: http://sawja.inria.fr/

3.5. Timbuk

KEYWORDS: Demonstration - Ocaml - Vérification de programmes - Tree Automata
FUNCTIONAL DESCRIPTION: Timbuk is a collection of tools for achieving proofs of reachability over Term Rewriting Systems and for manipulating Tree Automata (bottom-up non-deterministic finite tree automata)

RELEASE FUNCTIONAL DESCRIPTION: This version does no longer include the tree automata library but focuses on reachability analysis and equational approximations.

- Participant: Thomas Genet
- Contact: Thomas Genet
- URL: http://www.irisa.fr/celtique/genet/timbuk/

CONVECS Project-Team

5. New Software and Platforms

5.1. CADP Pro

Construction and Analysis of Distributed Processes
KEYWORDS: Formal methods - Verification
FUNCTIONAL DESCRIPTION: CADP (*Construction and Analysis of Distributed Processes* – formerly known as *CAESAR/ALDEBARAN Development Package*) [4] is a toolbox for protocols and distributed systems engineering.

In this toolbox, we develop and maintain the following tools:

- CAESAR.ADT [30] is a compiler that translates LOTOS abstract data types into C types and C functions. The translation involves pattern-matching compiling techniques and automatic recognition of usual types (integers, enumerations, tuples, etc.), which are implemented optimally.
- CAESAR [36], [35] is a compiler that translates LOTOS processes into either C code (for rapid prototyping and testing purposes) or finite graphs (for verification purposes). The translation is done using several intermediate steps, among which the construction of a Petri net extended with typed variables, data handling features, and atomic transitions.
- OPEN/CAESAR [31] is a generic software environment for developing tools that explore graphs on the fly (for instance, simulation, verification, and test generation tools). Such tools can be developed independently of any particular high level language. In this respect, OPEN/CAESAR plays a central role in CADP by connecting language-oriented tools with model-oriented tools. OPEN/CAESAR consists of a set of 16 code libraries with their programming interfaces, such as:
  - CAESAR_GRAPH, which provides the programming interface for graph exploration,
  - CAESAR_HASH, which contains several hash functions,
  - CAESAR_SOLVE, which resolves Boolean equation systems on the fly,
  - CAESAR_STACK, which implements stacks for depth-first search exploration, and
  - CAESAR_TABLE, which handles tables of states, transitions, labels, etc.

  A number of on-the-fly analysis tools have been developed within the OPEN/CAESAR environment, among which:
  - BISIMULATOR, which checks bisimulation equivalences and preorders,
  - CUNCTATOR, which performs steady-state simulation of continuous-time Markov chains,
  - DETERMINATOR, which eliminates stochastic nondeterminism in normal, probabilistic, or stochastic systems,
  - DISTRIBUTOR, which generates the graph of reachable states using several machines,
  - EVALUATOR, which evaluates MCL formulas,
  - EXECUTOR, which performs random execution,
  - EXHIBITOR, which searches for execution sequences matching a given regular expression,
  - GENERATOR, which constructs the graph of reachable states,
  - PROJECTOR, which computes abstractions of communicating systems,
  - REDUCTOR, which constructs and minimizes the graph of reachable states modulo various equivalence relations,

  - SIMULATOR, XSIMULATOR, and OCIS, which enable interactive simulation, and
  - TERMINATOR, which searches for deadlock states.

- BCG (*Binary Coded Graphs*) is both a file format for storing very large graphs on disk (using efficient compression techniques) and a software environment for handling this format. BCG also plays a key role in CADP as many tools rely on this format for their inputs/outputs. The BCG environment consists of various libraries with their programming interfaces, and of several tools, such as:
  - BCG_CMP, which compares two graphs,
  - BCG_DRAW, which builds a two-dimensional view of a graph,
  - BCG_EDIT, which allows the graph layout produced by BCG_DRAW to be modified interactively,
  - BCG_GRAPH, which generates various forms of practically useful graphs,
  - BCG_INFO, which displays various statistical information about a graph,
  - BCG_IO, which performs conversions between BCG and many other graph formats,
  - BCG_LABELS, which hides and/or renames (using regular expressions) the transition labels of a graph,
  - BCG_MIN, which minimizes a graph modulo strong or branching equivalences (and can also deal with probabilistic and stochastic systems),
  - BCG_STEADY, which performs steady-state numerical analysis of (extended) continuous-time Markov chains,
  - BCG_TRANSIENT, which performs transient numerical analysis of (extended) continuous-time Markov chains, and
  - XTL (*eXecutable Temporal Language*), which is a high level, functional language for programming exploration algorithms on BCG graphs. XTL provides primitives to handle states, transitions, labels, *successor* and *predecessor* functions, etc.

    For instance, one can define recursive functions on sets of states, which allow evaluation and diagnostic generation fixed point algorithms for usual temporal logics (such as HML [40], CTL [26], ACTL [28], etc.) to be defined in XTL.

- PBG (*Partitioned BCG Graph*) is a file format implementing the theoretical concept of *Partitioned LTS* [34] and providing a unified access to a graph partitioned in fragments distributed over a set of remote machines, possibly located in different countries. The PBG format is supported by several tools, such as:
  - PBG_CP, PBG_MV, and PBG_RM, which facilitate standard operations (copying, moving, and removing) on PBG files, maintaining consistency during these operations,
  - PBG_MERGE (formerly known as BCG_MERGE), which transforms a distributed graph into a monolithic one represented in BCG format,
  - PBG_INFO, which displays various statistical information about a distributed graph.

- The connection between explicit models (such as BCG graphs) and implicit models (explored on the fly) is ensured by OPEN/CAESAR-compliant compilers, e.g.:
  - BCG_OPEN, for models represented as BCG graphs,
  - CAESAR.OPEN, for models expressed as LOTOS descriptions,
  - EXP.OPEN, for models expressed as communicating automata,
  - FSP.OPEN, for models expressed as FSP [46] descriptions,
  - LNT.OPEN, for models expressed as LNT descriptions, and
  - SEQ.OPEN, for models represented as sets of execution traces.

The CADP toolbox also includes TGV (*Test Generation based on Verification*), which has been developed by the VERIMAG laboratory (Grenoble) and the VERTECS project-team at Inria Rennes – Bretagne-Atlantique.

The CADP tools are well-integrated and can be accessed easily using either the EUCALYPTUS graphical interface or the SVL [32] scripting language. Both EUCALYPTUS and SVL provide users with an easy and uniform access to the CADP tools by performing file format conversions automatically whenever needed and by supplying appropriate command-line options as the tools are invoked.

- Participants: Frédéric Lang, Hubert Garavel, Radu Mateescu and Wendelin Serwe
- Contact: Hubert Garavel
- URL: http://cadp.inria.fr/

5.2. TRAIAN

KEYWORDS: Compilation - LOTOS NT
FUNCTIONAL DESCRIPTION: TRAIAN is a compiler for translating LOTOS NT descriptions into C programs, which will be used for simulation, rapid prototyping, verification, and testing.

The current version of TRAIAN, which handles LOTOS NT types and functions only, has useful applications in compiler construction [33], being used in all recent compilers developed by CONVECS.

- Participants: Frédéric Lang, Hubert Garavel and Wendelin Serwe
- Contact: Hubert Garavel
- URL: http://convecs.inria.fr/software/traian/

DEDUCTEAM Project-Team

5. New Software and Platforms

5.1. Autotheo

KEYWORD: Automated deduction
SCIENTIFIC DESCRIPTION: Transformation of axiomatic theories into rewriting systems that can be used by iProverModulo.
FUNCTIONAL DESCRIPTION: Autotheo is a tool that transforms axiomatic theories into polarized rewriting systems, thus making them usable in iProverModulo. It supports several strategies to orient the axioms, some of them being proved to be complete, in the sense that ordered polarized resolution modulo the resulting systems is refutationally complete, some others being merely heuristics. In practice, Autotheo takes a TPTP input file and produces an input file for iProverModulo.
NEWS OF THE YEAR: Used by iProverModulo in its participation at the CASC-26 competition.

- Participant: Guillaume Burel
- Partner: ENSIIE
- Contact: Guillaume Burel
- Publication: Consistency Implies Cut Admissibility
- URL: http://www.ensiie.fr/~guillaume.burel/blackandwhite_autotheo.html.en

5.2. CoLoR

Coq Library on Rewriting and termination
KEYWORDS: Coq - Formalisation
FUNCTIONAL DESCRIPTION: CoLoR is a Coq library on rewriting theory and termination. It provides many definitions and theorems on various mathematical structures (quasi-ordered sets, relations, ordered semi-rings, etc.), data structures (lists, vectors, matrices, polynomials, finite graphs), term structures (strings, first-order terms, lambda-terms, etc.), transformation techniques (dependency pairs, semantic labeling, etc.) and (non-)termination criteria (polynomial and matrix interpretations, recursive path ordering, computability closure, etc.).
NEWS OF THE YEAR: 2017: Port to Coq 8.6 and 8.7.

- Authors: Frédéric Blanqui and Sébastien Hinderer
- Contact: Frédéric Blanqui
- Publications: CoLoR: a Coq library on well-founded rewrite relations and its application to the automated verification of termination certificates - Automated Verification of Termination Certificates - CoLoR: a Coq library on rewriting and termination
- URL: http://color.inria.fr/

5.3. Coqine

Coq In dEdukti
KEYWORDS: Higher-order logic - Formal methods - Proof
FUNCTIONAL DESCRIPTION: CoqInE is a plugin for the Coq software translating Coq proofs into Dedukti terms. It provides a Dedukti signature file faithfully encoding the underlying theory of Coq (or a sufficiently large subset of it). Current development is mostly focused on implementing support for Coq universe polymorphism. The generated ouput is meant to be type-checkable using the latest version of Dedukti.

- Contact: Guillaume Burel
- URL: http://www.ensiie.fr/~guillaume.burel/blackandwhite_coqInE.html.en

5.4. Dedukti

KEYWORD: Logical Framework
FUNCTIONAL DESCRIPTION: Dedukti is a proof-checker for the LambdaPi-calculus modulo. As it can be parametrized by an arbitrary set of rewrite rules, defining an equivalence relation, this calculus can express many different theories. Dedukti has been created for this purpose: to allow the interoperability of different theories.

Dedukti's core is based on the standard algorithm for type-checking semi-full pure type systems and implements a state-of-the-art reduction machine inspired from Matita's and modified to deal with rewrite rules.

Dedukti's input language features term declarations and definitions (opaque or not) and rewrite rule definitions. A basic module system allows the user to organize his project in different files and compile them separately.

Dedukti features matching modulo beta for a large class of patterns called Miller's patterns, allowing for more rewriting rules to be implemented in Dedukti.

- Participants: François Thiré, Gaspard Ferey, Guillaume Genestier and Rodolphe Lepigre
- Contact: François Thiré
- Publications: Dedukti:un vérificateur de preuves universel - Rewriting Modulo β in the $\lambda\Pi$-Calculus Modulo - Expressing theories in the $\lambda\Pi$-calculus modulo theory and in the Dedukti system
- URL: http://dedukti.gforge.inria.fr/

5.5. Holide

KEYWORD: Proof
FUNCTIONAL DESCRIPTION: Holide translates HOL proofs to Dedukti[OT] proofs, using the OpenTheory standard (common to HOL Light and HOL4). Dedukti[OT] being the encoding of OpenTheory in Dedukti.

- Contact: Guillaume Burel
- URL: http://deducteam.gforge.inria.fr/holide/

5.6. HOT

Higher-Order Termination
FUNCTIONAL DESCRIPTION: HOT is an automated termination prover for higher-order rewriting, based on the notion of computability closure.

- Contact: Frédéric Blanqui
- URL: http://rewriting.gforge.inria.fr/hot.html

5.7. iProver Modulo

KEYWORDS: Automated deduction - Automated theorem proving
SCIENTIFIC DESCRIPTION: Integration of ordered polarized resolution modulo theory into the prover iProver.
FUNCTIONAL DESCRIPTION: iProver Modulo is an extension of the automated theorem prover iProver originally developed by Konstantin Korovin at the University of Manchester. It implements ordered polarized resolution modulo theory, a refinement of the resolution method based on deduction modulo theory. It takes as input a proposition in predicate logic and a clausal rewriting system defining the theory in which the formula has to be proved. Normalization with respect to the term rewriting rules is performed very efficiently through translation into OCaml code, compilation and dynamic linking. Experiments have shown that ordered polarized resolution modulo dramatically improves proof search compared to using raw axioms.

NEWS OF THE YEAR: Participation at the automated-theorem-prover competition CASC-26 Integration of version 2.5 of iProver, adding support for types (TFF0)

- Participant: Guillaume Burel
- Partner: ENSIIE
- Contact: Guillaume Burel
- Publications: A Shallow Embedding of Resolution and Superposition Proofs into the ??-Calculus Modulo - Experimenting with deduction modulo
- URL: http://www.ensiie.fr/~guillaume.burel/blackandwhite_iProverModulo.html.en

5.8. mSAT

KEYWORD: Propositional logic
FUNCTIONAL DESCRIPTION: mSAT is a modular, proof-producing, SAT and SMT core based on Alt-Ergo Zero, written in OCaml. The solver accepts user-defined terms, formulas and theory, making it a good tool for experimenting. This tool produces resolution proofs as trees in which the leaves are user-defined proof of lemmas.

- Contact: Guillaume Bury
- Publication: mSAT:An OCaml SAT Solver
- URL: https://github.com/Gbury/mSAT

5.9. Rainbow

Termination certificate verifier
KEYWORDS: Demonstration - Code generation - Verification
FUNCTIONAL DESCRIPTION: Rainbow is a set of tools for automatically verifying the correctness of termination certificates expressed in the CPF format used in the annual international competition of termination tools. It contains: a tool xsd2coq for generating Coq data types for representing XML files valid with respect to some XML Schema, a tool xsd2ml for generating OCaml data types and functions for parsing XML files valid with respect to some XML Schema, a tool for translating a CPF file into a Coq script, and a standalone Coq certified tool for verifying the correctness of a CPF file.

- Author: Frédéric Blanqui
- Contact: Frédéric Blanqui
- Publications: Automated verification of termination certificates - Automated verification of termination certificates
- URL: http://color.inria.fr/rainbow.html

5.10. Krajono

KEYWORD: Proof
FUNCTIONAL DESCRIPTION: Krajono translates Matita proofs into Dedukti[CiC] (encoding of CiC in Dedukti) terms.

- Contact: François Thiré

5.11. archsat

KEYWORDS: Automated theorem proving - First-order logic - Propositional logic
FUNCTIONAL DESCRIPTION: Archsat is an automated theorem prover aimed at studying the integration of first-order theorem prover technologies, such as rewriting, into SMT solvers.

- Contact: Guillaume Bury
- URL: https://gforge.inria.fr/projects/archsat

GALLIUM Project-Team

6. New Software and Platforms

6.1. Compcert

The CompCert formally-verified C compiler
KEYWORDS: Compilers - Formal methods - Deductive program verification - C - Coq
FUNCTIONAL DESCRIPTION: CompCert is a compiler for the C programming language. Its intended use is the compilation of life-critical and mission-critical software written in C and meeting high levels of assurance. It accepts most of the ISO C 99 language, with some exceptions and a few extensions. It produces machine code for the ARM, PowerPC, RISC-V, and x86 architectures. What sets CompCert C apart from any other production compiler, is that it is formally verified to be exempt from miscompilation issues, using machine-assisted mathematical proofs (the Coq proof assistant). In other words, the executable code it produces is proved to behave exactly as specified by the semantics of the source C program. This level of confidence in the correctness of the compilation process is unprecedented and contributes to meeting the highest levels of software assurance. In particular, using the CompCert C compiler is a natural complement to applying formal verification techniques (static analysis, program proof, model checking) at the source code level: the correctness proof of CompCert C guarantees that all safety properties verified on the source code automatically hold as well for the generated executable.

RELEASE FUNCTIONAL DESCRIPTION: Novelties include a formally-verified type checker for CompCert C, a more careful modeling of pointer comparisons against the null pointer, algorithmic improvements in the handling of deeply nested struct and union types, much better ABI compatibility for passing composite values, support for GCC-style extended inline asm, and more complete generation of DWARF debugging information (contributed by AbsInt).

- Participants: Xavier Leroy, Sandrine Blazy, Jacques-Henri Jourdan, Sylvie Boldo and Guillaume Melquiond
- Partner: AbsInt Angewandte Informatik GmbH
- Contact: Xavier Leroy
- URL: http://compcert.inria.fr/

6.2. Diy

Do It Yourself
KEYWORD: Parallelism
FUNCTIONAL DESCRIPTION: The diy suite provides a set of tools for testing shared memory models: the litmus tool for running tests on hardware, various generators for producing tests from concise specifications, and herd, a memory model simulator. Tests are small programs written in x86, Power or ARM assembler that can thus be generated from concise specification, run on hardware, or simulated on top of memory models. Test results can be handled and compared using additional tools.

- Participants: Jade Alglave and Luc Maranget
- Partner: University College London UK
- Contact: Luc Maranget
- URL: http://diy.inria.fr/

6.3. Menhir

KEYWORDS: Compilation - Context-free grammars - Parsing

FUNCTIONAL DESCRIPTION: Menhir is a LR(1) parser generator for the OCaml programming language. That is, Menhir compiles LR(1) grammar specifications down to OCaml code. Menhir was designed and implemented by François Pottier and Yann Régis-Gianas.

- Contact: François Pottier
- Publications: A Simple, Possibly Correct LR Parser for C11 - Reachability and Error Diagnosis in LR(1) Parsers

6.4. OCaml

KEYWORDS: Functional programming - Static typing - Compilation
FUNCTIONAL DESCRIPTION: The OCaml language is a functional programming language that combines safety with expressiveness through the use of a precise and flexible type system with automatic type inference. The OCaml system is a comprehensive implementation of this language, featuring two compilers (a bytecode compiler, for fast prototyping and interactive use, and a native-code compiler producing efficient machine code for x86, ARM, PowerPC and System Z), a debugger, a documentation generator, a compilation manager, a package manager, and many libraries contributed by the user community.

- Participants: Damien Doligez, Xavier Leroy, Fabrice Le Fessant, Luc Maranget, Gabriel Scherer, Alain Frisch, Jacques Garrigue, Marc Shinwell, Jeremy Yallop and Leo White
- Contact: Damien Doligez
- URL: https://ocaml.org/

6.5. PASL

KEYWORD: Parallel computing
FUNCTIONAL DESCRIPTION: PASL is a C++ library for writing parallel programs targeting the broadly available multicore computers. The library provides a high level interface and can still guarantee very good efficiency and performance, primarily due to its scheduling and automatic granularity control mechanisms.

- Participants: Arthur Charguéraud, Michael Rainey and Umut Acar
- Contact: Michael Rainey
- URL: http://deepsea.inria.fr/pasl/

6.6. ZENON

FUNCTIONAL DESCRIPTION: Zenon is an automatic theorem prover based on the tableaux method. Given a first-order statement as input, it outputs a fully formal proof in the form of a Coq proof script. It has special rules for efficient handling of equality and arbitrary transitive relations. Although still in the prototype stage, it already gives satisfying results on standard automatic-proving benchmarks.

Zenon is designed to be easy to interface with front-end tools (for example integration in an interactive proof assistant), and also to be retargeted to output scripts for different frameworks (for example, Isabelle and Dedukti).

- Author: Damien Doligez
- Contact: Damien Doligez
- URL: http://zenon-prover.org/

6.7. OPAM Builder

KEYWORDS: Ocaml - Continuous integration - Opam

FUNCTIONAL DESCRIPTION: OPAM Builder checks in real-time the installability on a computer of all packages after any modification of the repository. To achieve this result, it uses smart mechanisms to compute incremental differencies between package updates, to be able to reuse cached compilations, and switch from a quadratic complexity to a linear complexity.

- Partner: OCamlPro
- Contact: Fabrice Le Fessant
- URL: http://github.com/OCamlPro/opam-builder

6.8. TLAPS

TLA+ proof system
KEYWORD: Proof assistant
FUNCTIONAL DESCRIPTION: TLAPS is a platform for developing and mechanically verifying proofs about TLA+ specifications. The TLA+ proof language is hierarchical and explicit, allowing a user to decompose the overall proof into proof steps that can be checked independently. TLAPS consists of a proof manager that interprets the proof language and generates a collection of proof obligations that are sent to backend verifiers. The current backends include the tableau-based prover Zenon for first-order logic, Isabelle/TLA+, an encoding of TLA+ set theory as an object logic in the logical framework Isabelle, an SMT backend designed for use with any SMT-lib compatible solver, and an interface to a decision procedure for propositional temporal logic.
NEWS OF THE YEAR: In 2017, we have continued to work on a complete reimplementation of the proof manager. One objective is a cleaner interaction with the TLA$^+$ front-ends, in particular SANY, the standard parser and semantic analyzer. The reimplementation is also necessary for extending the scope of the fragment of TLA$^+$ that is handled by TLAPS, in particular full temporal logic and module instantiation.

- Participants: Damien Doligez, Stephan Merz and Martin Riener
- Contact: Stephan Merz
- URL: https://tla.msr-inria.inria.fr/tlaps/content/Home.html

6.9. CFML

Interactive program verification using characteristic formulae
KEYWORDS: Coq - Software Verification - Deductive program verification - Separation Logic
FUNCTIONAL DESCRIPTION: The CFML tool supports the verification of OCaml programs through interactive Coq proofs. CFML proofs establish the full functional correctness of the code with respect to a specification. They may also be used to formally establish bounds on the asymptotic complexity of the code. The tool is made of two parts: on the one hand, a characteristic formula generator implemented as an OCaml program that parses OCaml code and produces Coq formulae, and, on the other hand, a Coq library that provides notations and tactics for manipulating characteristic formulae interactively in Coq.

- Participants: Arthur Charguéraud, Armaël Guéneau and François Pottier
- Contact: Arthur Charguéraud
- URL: http://www.charguéraud.org/softs/cfml/

6.10. ldrgen

Liveness-driven random C code generator
KEYWORDS: Code generation - Randomized algorithms - Static program analysis
FUNCTIONAL DESCRIPTION: The ldrgen program is a generator of C code: On every call it generates a new random C function and prints it to the standard output. The generator is "liveness-driven", which means that it tries to avoid generating dead code: All the computations it generates are (in a certain, limited sense) actually used to compute the function's return value. This is achieved by generating the program backwards, in combination with a simultaneous liveness analysis that guides the random generator's choices.

- Participant: Gergö Barany
- Contact: Gergö Barany
- Publication: Liveness-Driven Random Program Generation
- URL: https://github.com/gergo-/ldrgen

MARELLE Project-Team

5. New Software and Platforms

5.1. Coq

The Coq Proof Assistant
KEYWORDS: Proof - Certification - Formalisation
SCIENTIFIC DESCRIPTION: Coq is an interactive proof assistant based on the Calculus of (Co-)Inductive Constructions, extended with universe polymorphism. This type theory features inductive and co-inductive families, an impredicative sort and a hierarchy of predicative universes, making it a very expressive logic. The calculus allows to formalize both general mathematics and computer programs, ranging from theories of finite structures to abstract algebra and categories to programming language metatheory and compiler verification. Coq is organised as a (relatively small) kernel including efficient conversion tests on which are built a set of higher-level layers: a powerful proof engine and unification algorithm, various tactics/decision procedures, a transactional document model and, at the very top an IDE.
FUNCTIONAL DESCRIPTION: Coq provides both a dependently-typed functional programming language and a logical formalism, which, altogether, support the formalisation of mathematical theories and the specification and certification of properties of programs. Coq also provides a large and extensible set of automatic or semi-automatic proof methods. Coq's programs are extractible to OCaml, Haskell, Scheme, ...

RELEASE FUNCTIONAL DESCRIPTION: Version 8.7 features a large amount of work on cleaning and speeding up the code base, notably the work of Pierre-Marie Pédrot on making the tactic-level system insensitive to existential variable expansion, providing a safer API to plugin writers and making the code more robust.

New tactics: Variants of tactics supporting existential variables "eassert", "eenough", etc. by Hugo Herbelin. Tactics "extensionality in H" and "inversion_sigma" by Jason Gross, "specialize with" accepting partial bindings by Pierre Courtieu.

Cumulative Polymorphic Inductive Types, allowing cumulativity of universes to go through applied inductive types, by Amin Timany and Matthieu Sozeau.

The SSReflect plugin by Georges Gonthier, Assia Mahboubi and Enrico Tassi was integrated (with its documentation in the reference manual) by Maxime Dénès, Assia Mahboubi and Enrico Tassi.

The "coq_makefile" tool was completely redesigned to improve its maintainability and the extensibility of generated Makefiles, and to make "_CoqProject" files more palatable to IDEs by Enrico Tassi.

A lot of other changes are described in the CHANGES file.
NEWS OF THE YEAR: Version 8.7 was released in October 2017 and version 8.7.1 in December 2017, development started in January 2017. This is the second release of Coq developed on a time-based development cycle. Its development spanned 9 months from the release of Coq 8.6 and was based on a public road-map. It attracted many external contributions. Code reviews and continuous integration testing were systematically used before integration of new features, with an important focus given to compatibility and performance issues.

The main scientific advance in this version is the integration of cumulative inductive types in the system. More practical advances in stability, performance, usability and expressivity of tactics were also implemented, resulting in a mostly backwards-compatible but appreciably faster and more robust release. Much work on plugin extensions to Coq by the same development team has also been going on in parallel, including work on JSCoq by Emilio JG Arias, Ltac 2 by P.M-Pédrot, which required synchronised changes of the main codebase. In 2017, the construction of the Coq Consortium by Yves Bertot and Maxime Dénès has greatly advanced and is now nearing its completion.

- Participants: Abhishek Anand, C. J. Bell, Yves Bertot, Frédéric Besson, Tej Chajed, Pierre Courtieu, Maxime Denes, Julien Forest, Emilio Jesús Gallego Arias, Gaëtan Gilbert, Benjamin Grégoire, Jason Gross, Hugo Herbelin, Ralf Jung, Matej Kosik, Sam Pablo Kuper, Xavier Leroy, Pierre Letouzey, Assia Mahboubi, Cyprien Mangin, Érik Martin-Dorel, Olivier Marty, Guillaume Melquiond, Pierre-Marie Pédrot, Benjamin C. Pierce, Lars Rasmusson, Yann Régis-Gianas, Lionel Rieg, Valentin Robert, Thomas Sibut-Pinote, Michael Soegtrop, Matthieu Sozeau, Arnaud Spiwack, Paul Steckler, George Stelle, Pierre-Yves Strub, Enrico Tassi, Hendrik Tews, Laurent Théry, Amin Timany, Vadim Zaliva and Théo Zimmermann
- Partners: CNRS - Université Paris-Sud - ENS Lyon - Université Paris-Diderot
- Contact: Matthieu Sozeau
- Publication: The Coq Proof Assistant, version 8.7.1
- URL: http://coq.inria.fr/

5.2. Easycrypt

FUNCTIONAL DESCRIPTION: EasyCrypt is a toolset for reasoning about relational properties of probabilistic computations with adversarial code. Its main application is the construction and verification of game-based cryptographic proofs. EasyCrypt can also be used for reasoning about differential privacy.

- Participants: Benjamin Grégoire, Gilles Barthe and Pierre-Yves Strub
- Contact: Gilles Barthe
- URL: https://www.easycrypt.info/trac/

5.3. ELPI

Embeddable Lambda Prolog Interpreter
KEYWORDS: Constraint Programming - Programming language - Higher-order logic
FUNCTIONAL DESCRIPTION: ELPI is a lambdaProlog interpreter written in OCaml, easy to embed in software written in the same language.

- Contact: Enrico Tassi

5.4. Math-Components

Mathematical Components library
FUNCTIONAL DESCRIPTION: The Mathematical Components library is a set of Coq libraries that cover the mechanization of the proof of the Odd Order Theorem.

RELEASE FUNCTIONAL DESCRIPTION: The library includes 16 more theory files, covering in particular field and Galois theory, advanced character theory, and a construction of algebraic numbers.

- Participants: Alexey Solovyev, Andrea Asperti, Assia Mahboubi, Cyril Cohen, Enrico Tassi, François Garillot, Georges Gonthier, Ioana Pasca, Jeremy Avigad, Laurence Rideau, Laurent Théry, Russell O'Connor, Sidi Ould Biha, Stéphane Le Roux and Yves Bertot
- Contact: Assia Mahboubi
- URL: http://math-comp.github.io/math-comp/

5.5. Semantics

KEYWORDS: Semantic - Programming language - Coq
FUNCTIONAL DESCRIPTION: A didactical Coq development to introduce various semantics styles. Shows how to derive an interpreter, a verifier, or a program analyser from formal descriptions, and how to prove their consistency.

This is a library for the Coq system, where the description of a toy programming language is presented. The value of this library is that it can be re-used in classrooms to teach programming language semantics or the Coq system. The topics covered include introductory notions to domain theory, pre and post-conditions, abstract interpretation, and the proofs of consistency between all these point of views on the same programming language. Standalone tools for the object programming language can be derived from this development.

- Participants: Christine Paulin and Yves Bertot
- Contact: Yves Bertot
- URL: http://www-sop.inria.fr/members/Yves.Bertot/proofs/semantics_survey.tgz

5.6. Ssreflect

FUNCTIONAL DESCRIPTION: Ssreflect is a tactic language extension to the Coq system, developed by the Mathematical Components team.

- Participants: Assia Mahboubi, Cyril Cohen, Enrico Tassi, Georges Gonthier, Laurence Rideau, Laurent Théry and Yves Bertot
- Contact: Yves Bertot
- URL: http://math-comp.github.io/math-comp/

5.7. AutoGnP

KEYWORDS: Formal methods - Security - Cryptography
FUNCTIONAL DESCRIPTION: autoGnP is an automated tool for analyzing the security of padding-based public-key encryption schemes (i.e. schemes built from trapdoor permutations and hash functions). This years we extended the tool to be able to deal with schemes based on cyclic groups and bilinear maps.

- Participants: Benjamin Grégoire, Gilles Barthe and Pierre-Yves Strub
- Contact: Gilles Barthe
- URL: https://github.com/ZooCrypt/AutoGnP

MEXICO Project-Team

6. New Software and Platforms

6.1. COSMOS

FUNCTIONAL DESCRIPTION: COSMOS is a statistical model checker for the Hybrid Automata Stochastic Logic (HASL). HASL employs Linear Hybrid Automata (LHA), a generalization of Deterministic Timed Automata (DTA), to describe accepting execution paths of a Discrete Event Stochastic Process (DESP), a class of stochastic models which includes, but is not limited to, Markov chains. As a result HASL verification turns out to be a unifying framework where sophisticated temporal reasoning is naturally blended with elaborate reward-based analysis. COSMOS takes as input a DESP (described in terms of a Generalized Stochastic Petri Net), an LHA and an expression Z representing the quantity to be estimated. It returns a confidence interval estimation of Z, recently, it has been equipped with functionalities for rare event analysis. COSMOS is written in C++

- Participants: Benoît Barbot, Hilal Djafri, Marie Duflot-Kremer, Paolo Ballarini and Serge Haddad
- Contact: Hilal Djafri
- URL: http://www.lsv.ens-cachan.fr/~barbot/cosmos/

6.2. CosyVerif

FUNCTIONAL DESCRIPTION: CosyVerif is a platform dedicated to the formal specification and verification of dynamic systems. It allows to specify systems using several formalisms (such as automata and Petri nets), and to run verification tools on these models.

- Participants: Alban Linard, Fabrice Kordon, Laure Petrucci and Serge Haddad
- Partners: LIP6 - LSV - LIPN (Laboratoire d'Informatique de l'Université Paris Nord)
- Contact: Serge Haddad
- URL: http://www.cosyverif.org/

6.3. Mole

FUNCTIONAL DESCRIPTION: Mole computes, given a safe Petri net, a finite prefix of its unfolding. It is designed to be compatible with other tools, such as PEP and the Model-Checking Kit, which are using the resulting unfolding for reachability checking and other analyses. The tool Mole arose out of earlier work on Petri nets.

- Participant: Stefan Schwoon
- Contact: Stefan Schwoon
- URL: http://www.lsv.ens-cachan.fr/~schwoon/tools/mole/

PARSIFAL Project-Team

5. New Software and Platforms

5.1. Abella

FUNCTIONAL DESCRIPTION: Abella is an interactive theorem prover for reasoning about computations given as relational specifications. Abella is particuarly well suited for reasoning about binding constructs.

- Participants: Dale Miller, Gopalan Nadathur, Kaustuv Chaudhuri, Mary Southern, Matteo Cimini, Olivier Savary-Bélanger and Yuting Wang
- Partner: Department of Computer Science and Engineering, University of Minnesota
- Contact: Kaustuv Chaudhuri
- URL: http://abella-prover.org/

5.2. Bedwyr

Bedwyr - A proof search approach to model checking
FUNCTIONAL DESCRIPTION: Bedwyr is a generalization of logic programming that allows model checking directly on syntactic expressions that possibly contain bindings. This system, written in OCaml, is a direct implementation of two recent advances in the theory of proof search.

It is possible to capture both finite success and finite failure in a sequent calculus. Proof search in such a proof system can capture both may and must behavior in operational semantics. Higher-order abstract syntax is directly supported using term-level lambda-binders, the nabla quantifier, higher-order pattern unification, and explicit substitutions. These features allow reasoning directly on expressions containing bound variables.

The distributed system comes with several example applications, including the finite pi-calculus (operational semantics, bisimulation, trace analyses, and modal logics), the spi-calculus (operational semantics), value-passing CCS, the lambda-calculus, winning strategies for games, and various other model checking problems.

- Participants: Dale Miller, Quentin Heath and Roberto Blanco Martinez
- Contact: Quentin Heath
- URL: http://slimmer.gforge.inria.fr/bedwyr/

5.3. Checkers

Checkers - A proof verifier
KEYWORDS: Proof - Certification - Verification
FUNCTIONAL DESCRIPTION: Checkers is a tool in Lambda-prolog for the certification of proofs. Checkers consists of a kernel which is based on LKF and is based on the notion of ProofCert.

- Participants: Giselle Machado Nogueira Reis, Marco Volpe and Tomer Libal
- Contact: Tomer Libal
- URL: https://github.com/proofcert/checkers

5.4. Psyche

Proof-Search factorY for Collaborative HEuristics
FUNCTIONAL DESCRIPTION: Psyche is a modular platform for automated or interactive theorem proving, programmed in OCaml and built on an architecture (similar to LCF) where a trusted kernel interacts with plugins. The kernel offers an API of proof-search primitives, and plugins are programmed on top of the API to implement search strategies. This architecture is set up for pure logical reasoning as well as for theory-specific reasoning, for various theories.

RELEASE FUNCTIONAL DESCRIPTION: It is now equipped with the machinery to handle quantifiers and quantifier-handling techniques. Concretely, it uses meta-variables to delay the instantiation of existential variables, and constraints on meta-variables are propagated through the various branches of the search-space, in a way that allows local backtracking. The kernel, of about 800 l.o.c., is purely functional.

- Participants: Assia Mahboubi, Jean-Marc Notin and Stéphane Graham-Lengrand
- Contact: Stéphane Graham-Lengrand
- URL: http://www.lix.polytechnique.fr/~lengrand/Psyche/

PI.R2 Project-Team

5. New Software and Platforms

5.1. Coq

The Coq Proof Assistant
KEYWORDS: Proof - Certification - Formalisation
SCIENTIFIC DESCRIPTION: Coq is an interactive proof assistant based on the Calculus of (Co-)Inductive Constructions, extended with universe polymorphism. This type theory features inductive and co-inductive families, an impredicative sort and a hierarchy of predicative universes, making it a very expressive logic. The calculus allows to formalize both general mathematics and computer programs, ranging from theories of finite structures to abstract algebra and categories to programming language metatheory and compiler verification. Coq is organised as a (relatively small) kernel including efficient conversion tests on which are built a set of higher-level layers: a powerful proof engine and unification algorithm, various tactics/decision procedures, a transactional document model and, at the very top an IDE.
FUNCTIONAL DESCRIPTION: Coq provides both a dependently-typed functional programming language and a logical formalism, which, altogether, support the formalisation of mathematical theories and the specification and certification of properties of programs. Coq also provides a large and extensible set of automatic or semi-automatic proof methods. Coq's programs are extractible to OCaml, Haskell, Scheme, ...

RELEASE FUNCTIONAL DESCRIPTION: Version 8.7 features a large amount of work on cleaning and speeding up the code base, notably the work of Pierre-Marie Pédrot on making the tactic-level system insensitive to existential variable expansion, providing a safer API to plugin writers and making the code more robust.

New tactics: Variants of tactics supporting existential variables "eassert", "eenough", etc. by Hugo Herbelin. Tactics "extensionality in H" and "inversion_sigma" by Jason Gross, "specialize with" accepting partial bindings by Pierre Courtieu.

Cumulative Polymorphic Inductive Types, allowing cumulativity of universes to go through applied inductive types, by Amin Timany and Matthieu Sozeau.

The SSReflect plugin by Georges Gonthier, Assia Mahboubi and Enrico Tassi was integrated (with its documentation in the reference manual) by Maxime Dénès, Assia Mahboubi and Enrico Tassi.

The "coq_makefile" tool was completely redesigned to improve its maintainability and the extensibility of generated Makefiles, and to make "_CoqProject" files more palatable to IDEs by Enrico Tassi.

A lot of other changes are described in the CHANGES file.
NEWS OF THE YEAR: Version 8.7 was released in October 2017 and version 8.7.1 in December 2017, development started in January 2017. This is the second release of Coq developed on a time-based development cycle. Its development spanned 9 months from the release of Coq 8.6 and was based on a public road-map. It attracted many external contributions. Code reviews and continuous integration testing were systematically used before integration of new features, with an important focus given to compatibility and performance issues.

The main scientific advance in this version is the integration of cumulative inductive types in the system. More practical advances in stability, performance, usability and expressivity of tactics were also implemented, resulting in a mostly backwards-compatible but appreciably faster and more robust release. Much work on plugin extensions to Coq by the same development team has also been going on in parallel, including work on JSCoq by Emilio JG Arias, Ltac 2 by P.M-Pédrot, which required synchronised changes of the main codebase. In 2017, the construction of the Coq Consortium by Yves Bertot and Maxime Dénès has greatly advanced and is now nearing its completion.

- Participants: Abhishek Anand, C. J. Bell, Yves Bertot, Frédéric Besson, Tej Chajed, Pierre Courtieu, Maxime Denes, Julien Forest, Emilio Jesús Gallego Arias, Gaëtan Gilbert, Benjamin Grégoire, Jason Gross, Hugo Herbelin, Ralf Jung, Matej Kosik, Sam Pablo Kuper, Xavier Leroy, Pierre Letouzey, Assia Mahboubi, Cyprien Mangin, Érik Martin-Dorel, Olivier Marty, Guillaume Melquiond, Pierre-Marie Pédrot, Benjamin C. Pierce, Lars Rasmusson, Yann Régis-Gianas, Lionel Rieg, Valentin Robert, Thomas Sibut-Pinote, Michael Soegtrop, Matthieu Sozeau, Arnaud Spiwack, Paul Steckler, George Stelle, Pierre-Yves Strub, Enrico Tassi, Hendrik Tews, Laurent Théry, Amin Timany, Vadim Zaliva and Théo Zimmermann
- Partners: CNRS - Université Paris-Sud - ENS Lyon - Université Paris-Diderot
- Contact: Matthieu Sozeau
- Publication: The Coq Proof Assistant, version 8.7.1
- URL: http://coq.inria.fr/

5.2. Equations

KEYWORDS: Coq - Dependent Pattern-Matching - Proof assistant - Functional programming
SCIENTIFIC DESCRIPTION: Equations is a tool designed to help with the definition of programs in the setting of dependent type theory, as implemented in the Coq proof assistant. Equations provides a syntax for defining programs by dependent pattern-matching and well-founded recursion and compiles them down to the core type theory of Coq, using the primitive eliminators for inductive types, accessibility and equality. In addition to the definitions of programs, it also automatically derives useful reasoning principles in the form of propositional equations describing the functions, and an elimination principle for calls to this function. It realizes this using a purely definitional translation of high-level definitions to core terms, without changing the core calculus in any way, or using axioms.
FUNCTIONAL DESCRIPTION: Equations is a function definition plugin for Coq (supporting Coq 8.6 and 8.7), that allows the definition of functions by dependent pattern-matching and well-founded, mutual or nested structural recursion and compiles them into core terms. It automatically derives the clauses equations, the graph of the function and its associated elimination principle.

Equations is based on a simplification engine for the dependent equalities appearing in dependent eliminations that is also usable as a separate tactic, providing an axiom-free variant of dependent destruction. The main features of Equations include:

Dependent pattern-matching in the style of Agda/Epigram, with inaccessible patterns, with and where clauses. The use of the K axiom or a proof of K is configurable.

Support for well-founded recursion using by rec annotations, and automatic derivation of the subterm relation for inductive families.

Support for mutual and nested structural recursion using with and where auxilliary definitions, allowing to factor multiple uses of the same nested fixpoint definition. It proves the expected elimination principles for mutual and nested definitions.

Automatic generation of the defining equations as rewrite rules for every definition.

Automatic generation of the unfolding lemma for well-founded definitions (requiring only functional extensionality).

Automatic derivation of the graph of the function and its elimination principle. In case the automation fails to prove these principles, the user is asked to provide a proof.

A new dependent elimination tactic based on the same splitting tree compilation scheme that can advantageously replace dependent destruction and sometimes inversion as well. The as clause of dependent elimination allows to specify exactly the patterns and naming of new variables needed for an elimination.

A set of Derive commands for automatic derivation of constructions from an inductive type: its signature, no-confusion property, well-founded subterm relation and decidable equality proof, if applicable.
NEWS OF THE YEAR: Equations 1.0 was released in december this year, after 7 years of (non-continuous) development. It provides the first feature-full version of the software. It has been tried and tested on small to medium scale examples (available on the website). Equations was presented at the Type Theory Tools EUTypes meeting in January 2017 in Paris, and another demo/presentation will be given at PEPM 2018 in Los Angeles in January 2018.

- Participants: Matthieu Sozeau and Cyprien Mangin
- Contact: Matthieu Sozeau
- Publications: Equations reloaded - Equations for Hereditary Substitution in Leivant's Predicative System F: A Case Study - Equations: A Dependent Pattern-Matching Compiler
- URL: http://mattam82.github.io/Coq-Equations/

SUMO Project-Team

6. New Software and Platforms

6.1. Active Workspaces

KEYWORDS: Active workspace - Collaborative systems - Artifact centric workflow system
SCIENTIFIC DESCRIPTION: Tool for computer supported cooperative work where a user's workspace is given by an active structured repository containing the pending tasks together with information needed to perform the tasks. Communication between active workspaces is asynchronous using message passing. The tool is based on the model of guarded attribute grammars.

- Authors: Éric Badouel and Robert Nsaibirni
- Contact: Éric Badouel
- URL: http://people.rennes.inria.fr/Eric.Badouel/Research/ActiveWorkspaces.html

6.2. DAXML

KEYWORDS: XML - Web Services - Distributed Software - Active documents
SCIENTIFIC DESCRIPTION: DAXML is an interpreter and implementation of Distributed Active Documents, a formalism for data centric design of Web Services. This implementation is based on a REST framework, and can run on a network of machines connected to internet and equiped with JAVA.
FUNCTIONAL DESCRIPTION: This prototype interprets distributed Active XML documents. It can be used to deploy services defined as active documents over the web.

- Participants: Benoît Masson and Loïc Hélouët
- Contact: Loïc Hélouët
- URL: http://www.irisa.fr/sumo/Software/DAXML/

6.3. Sigali

FUNCTIONAL DESCRIPTION: Sigali is a model-checking tool that operates on ILTS (Implicit Labeled Transition Systems, an equational representation of an automaton), an intermediate model for discrete event systems. It offers functionalities for verification of reactive systems and discrete controller synthesis. The techniques used consist in manipulating the system of equations instead of the set of solutions, which avoids the enumeration of the state space. Each set of states is uniquely characterized by a predicate and the operations on sets can be equivalently performed on the associated predicates. Therefore, a wide spectrum of properties, such as liveness, invariance, reachability and attractivity, can be checked. Algorithms for the computation of predicates on states are also available. Sigali is connected with the Polychrony environment (Tea project-team) as well as the Matou environment (VERIMAG), thus allowing the modeling of reactive systems by means of Signal Specification or Mode Automata and the visualization of the synthesized controller by an interactive simulation of the controlled system.

- Contact: Hervé Marchand

6.4. SIMSTORS

Simulator for stochastic regulated systems
KEYWORDS: Simulation - Public transport - Stochastic models - Distributed systems

FUNCTIONAL DESCRIPTION: SIMSTORS is a software for the simulation of stochastic concurrent timed systems. The heart of the software is a variant of stochastic and timed Petri nets, whose execution is controlled by a regulation policy (a controller), or a predetermined theoretical schedule. The role of the regulation policy is to control the system to realize obbjectives or a schedule when it exists with the best possible precision. SIMSTORS is well adapted to represent systems with randomness, parallelism, tasks scheduling, and resources. It is currently in use within collaboration P22 with Asltom Transport, where it is used to model metro traffic and evaluate performance of regulation solutions. This software allows for step by step simulation, but also for efficient performance analysis of systems such as production cells or train systems. The initial implementation was released in 2015, and the software is protected by the APP.

In 2017, SIMSTORS has been extended along two main axes: on one hand, SIMSTORS models were extended to handle situations where shared resources can be occupied by more than one object (this is of paramount importance to represent conveyors, roads occupied by cars, or train tracks with smoothed scheduling allowing shared sections among trains) with priorities, constraint on their ordering and individual characteristics. This allows for instance to model vehicles with different speeds on a road, while handling safety distance constraints. On the other hand, SIMSTORS models were extended to allow control of stochastic nets based on decision rules that follow optimization schemes.

- Participants: Abd El Karim Kecir and Loïc Hélouët
- Contact: Loïc Hélouët
- URL: http://www.irisa.fr/sumo/Software/SIMSTORS/

6.5. Tipex

TImed Properties Enforcement during eXecution
KEYWORDS: Monitoring - Controller synthesis - Formal methods
FUNCTIONAL DESCRIPTION: We are implementing a prototype tool named Tipex (TImed Properties Enforcement during eXecution) for the enforcement of timed properties. Tipex is based on the theory and algorithms that we develop for the synthesis of enforcement monitors for properties specified by timed automata (TA). The prototype is developped in python, and uses the PyUPPAAL and DBMpyuppaal libraries of the UPPAAL tool . It is currently restricted to safety and co-safety timed property. The property provided as input to the tool is a TA that can be specified using the UPPAAL tool, and is stored in XML format. The tool synthesizes an enforcement monitor from this TA, which can then be used to enforce a sequence of timed events to satisfy the property. Experiments have been conducted on a set of case studies. This allowed to validate the architecture and feasibility of enforcement monitoring in a timed setting and to have a first assessment of performance (and to what extent the overhead induced by monitoring is negligible).

- Participants: Thierry Jéron, Srinivas Pinisetty and Hervé Marchand
- Contact: Thierry Jéron

TOCCATA Project-Team

6. New Software and Platforms

6.1. Alt-Ergo

Automated theorem prover for software verification
KEYWORDS: Software Verification - Automated theorem proving
FUNCTIONAL DESCRIPTION: Alt-Ergo is an automatic solver of formulas based on SMT technology. It is especially designed to prove mathematical formulas generated by program verification tools, such as Frama-C for C programs, or SPARK for Ada code. Initially developed in Toccata research team, Alt-Ergo's distribution and support are provided by OCamlPro since September 2013.

RELEASE FUNCTIONAL DESCRIPTION: the "SAT solving" part can now be delegated to an external plugin, new experimental SAT solver based on mini-SAT, provided as a plugin. This solver is, in general, more efficient on ground problems, heuristics simplification in the default SAT solver and in the matching (instantiation) module, re-implementation of internal literals representation, improvement of theories combination architecture, rewriting some parts of the formulas module, bugfixes in records and numbers modules, new option "-no-Ematching" to perform matching without equality reasoning (i.e. without considering "equivalence classes"). This option is very useful for benchmarks coming from Atelier-B, two new experimental options: "-save-used-context" and "-replay-used-context". When the goal is proved valid, the first option allows to save the names of useful axioms into a ".used" file. The second one is used to replay the proof using only the axioms listed in the corresponding ".used" file. Note that the replay may fail because of the absence of necessary ground terms generated by useless axioms (that are not included in .used file) during the initial run.

- Participants: Alain Mebsout, Évelyne Contejean, Mohamed Iguernelala, Stéphane Lescuyer and Sylvain Conchon
- Partner: OCamlPro
- Contact: Sylvain Conchon
- URL: http://alt-ergo.lri.fr

6.2. CFML

Interactive program verification using characteristic formulae
KEYWORDS: Coq - Software Verification - Deductive program verification - Separation Logic
FUNCTIONAL DESCRIPTION: The CFML tool supports the verification of OCaml programs through interactive Coq proofs. CFML proofs establish the full functional correctness of the code with respect to a specification. They may also be used to formally establish bounds on the asymptotic complexity of the code. The tool is made of two parts: on the one hand, a characteristic formula generator implemented as an OCaml program that parses OCaml code and produces Coq formulae, and, on the other hand, a Coq library that provides notations and tactics for manipulating characteristic formulae interactively in Coq.

- Participants: Arthur Charguéraud, Armaël Guéneau and François Pottier
- Contact: Arthur Charguéraud
- URL: http://www.charguéraud.org/softs/cfml/

6.3. Coq

The Coq Proof Assistant
KEYWORDS: Proof - Certification - Formalisation

SCIENTIFIC DESCRIPTION: Coq is an interactive proof assistant based on the Calculus of (Co-)Inductive Constructions, extended with universe polymorphism. This type theory features inductive and co-inductive families, an impredicative sort and a hierarchy of predicative universes, making it a very expressive logic. The calculus allows to formalize both general mathematics and computer programs, ranging from theories of finite structures to abstract algebra and categories to programming language metatheory and compiler verification. Coq is organised as a (relatively small) kernel including efficient conversion tests on which are built a set of higher-level layers: a powerful proof engine and unification algorithm, various tactics/decision procedures, a transactional document model and, at the very top an IDE.
FUNCTIONAL DESCRIPTION: Coq provides both a dependently-typed functional programming language and a logical formalism, which, altogether, support the formalisation of mathematical theories and the specification and certification of properties of programs. Coq also provides a large and extensible set of automatic or semi-automatic proof methods. Coq's programs are extractible to OCaml, Haskell, Scheme, ...

RELEASE FUNCTIONAL DESCRIPTION: Version 8.7 features a large amount of work on cleaning and speeding up the code base, notably the work of Pierre-Marie Pédrot on making the tactic-level system insensitive to existential variable expansion, providing a safer API to plugin writers and making the code more robust.

New tactics: Variants of tactics supporting existential variables "eassert", "eenough", etc. by Hugo Herbelin. Tactics "extensionality in H" and "inversion_sigma" by Jason Gross, "specialize with" accepting partial bindings by Pierre Courtieu.

Cumulative Polymorphic Inductive Types, allowing cumulativity of universes to go through applied inductive types, by Amin Timany and Matthieu Sozeau.

The SSReflect plugin by Georges Gonthier, Assia Mahboubi and Enrico Tassi was integrated (with its documentation in the reference manual) by Maxime Dénès, Assia Mahboubi and Enrico Tassi.

The "coq_makefile" tool was completely redesigned to improve its maintainability and the extensibility of generated Makefiles, and to make "_CoqProject" files more palatable to IDEs by Enrico Tassi.

A lot of other changes are described in the CHANGES file.
NEWS OF THE YEAR: Version 8.7 was released in October 2017 and version 8.7.1 in December 2017, development started in January 2017. This is the second release of Coq developed on a time-based development cycle. Its development spanned 9 months from the release of Coq 8.6 and was based on a public road-map. It attracted many external contributions. Code reviews and continuous integration testing were systematically used before integration of new features, with an important focus given to compatibility and performance issues.

The main scientific advance in this version is the integration of cumulative inductive types in the system. More practical advances in stability, performance, usability and expressivity of tactics were also implemented, resulting in a mostly backwards-compatible but appreciably faster and more robust release. Much work on plugin extensions to Coq by the same development team has also been going on in parallel, including work on JSCoq by Emilio JG Arias, Ltac 2 by P.M-Pédrot, which required synchronised changes of the main codebase. In 2017, the construction of the Coq Consortium by Yves Bertot and Maxime Dénès has greatly advanced and is now nearing its completion.

- Participants: Abhishek Anand, C. J. Bell, Yves Bertot, Frédéric Besson, Tej Chajed, Pierre Courtieu, Maxime Denes, Julien Forest, Emilio Jesús Gallego Arias, Gaëtan Gilbert, Benjamin Grégoire, Jason Gross, Hugo Herbelin, Ralf Jung, Matej Kosik, Sam Pablo Kuper, Xavier Leroy, Pierre Letouzey, Assia Mahboubi, Cyprien Mangin, Érik Martin-Dorel, Olivier Marty, Guillaume Melquiond, Pierre-Marie Pédrot, Benjamin C. Pierce, Lars Rasmusson, Yann Régis-Gianas, Lionel Rieg, Valentin Robert, Thomas Sibut-Pinote, Michael Soegtrop, Matthieu Sozeau, Arnaud Spiwack, Paul Steckler, George Stelle, Pierre-Yves Strub, Enrico Tassi, Hendrik Tews, Laurent Théry, Amin Timany, Vadim Zaliva and Théo Zimmermann
- Partners: CNRS - Université Paris-Sud - ENS Lyon - Université Paris-Diderot
- Contact: Matthieu Sozeau
- Publication: The Coq Proof Assistant, version 8.7.1
- URL: http://coq.inria.fr/

6.4. CoqInterval

Interval package for Coq
KEYWORDS: Interval arithmetic - Coq
FUNCTIONAL DESCRIPTION: CoqInterval is a library for the proof assistant Coq.

It provides several tactics for proving theorems on enclosures of real-valued expressions. The proofs are performed by an interval kernel which relies on a computable formalization of floating-point arithmetic in Coq.

The Marelle team developed a formalization of rigorous polynomial approximation using Taylor models in Coq. In 2014, this library has been included in CoqInterval.

- Participants: Assia Mahboubi, Érik Martin-Dorel, Guillaume Melquiond, Jean-Michel Muller, Laurence Rideau, Laurent Théry, Micaela Mayero, Mioara Joldes, Nicolas Brisebarre and Thomas Sibut-Pinote
- Contact: Guillaume Melquiond
- Publications: Proving bounds on real-valued functions with computations - Floating-point arithmetic in the Coq system - Proving Tight Bounds on Univariate Expressions with Elementary Functions in Coq - Formally Verified Approximations of Definite Integrals - Formally Verified Approximations of Definite Integrals
- URL: http://coq-interval.gforge.inria.fr/

6.5. Coquelicot

The Coquelicot library for real analysis in Coq
KEYWORDS: Coq - Real analysis
FUNCTIONAL DESCRIPTION: Coquelicot is library aimed for supporting real analysis in the Coq proof assistant. It is designed with three principles in mind. The first is the user-friendliness, achieved by implementing methods of automation, but also by avoiding dependent types in order to ease the stating and readability of theorems. This latter part was achieved by defining total function for basic operators, such as limits or integrals. The second principle is the comprehensiveness of the library. By experimenting on several applications, we ensured that the available theorems are enough to cover most cases. We also wanted to be able to extend our library towards more generic settings, such as complex analysis or Euclidean spaces. The third principle is for the Coquelicot library to be a conservative extension of the Coq standard library, so that it can be easily combined with existing developments based on the standard library.

- Participants: Catherine Lelay, Guillaume Melquiond and Sylvie Boldo
- Contact: Sylvie Boldo
- URL: http://coquelicot.saclay.inria.fr/

6.6. Cubicle

The Cubicle model checker modulo theories
KEYWORDS: Model Checking - Software Verification
FUNCTIONAL DESCRIPTION: Cubicle is an open source model checker for verifying safety properties of array-based systems, which corresponds to a syntactically restricted class of parametrized transition systems with states represented as arrays indexed by an arbitrary number of processes. Cache coherence protocols and mutual exclusion algorithms are typical examples of such systems.

- Participants: Alain Mebsout and Sylvain Conchon
- Contact: Sylvain Conchon
- URL: http://cubicle.lri.fr/

6.7. Flocq

The Flocq library for formalizing floating-point arithmetic in Coq
KEYWORDS: Floating-point - Arithmetic code - Coq
FUNCTIONAL DESCRIPTION: The Flocq library for the Coq proof assistant is a comprehensive formalization of floating-point arithmetic: core definitions, axiomatic and computational rounding operations, high-level properties. It provides a framework for developers to formally verify numerical applications.

Flocq is currently used by the CompCert verified compiler to support floating-point computations.

- Participants: Guillaume Melquiond, Pierre Roux and Sylvie Boldo
- Contact: Sylvie Boldo
- Publications: Flocq: A Unified Library for Proving Floating-point Algorithms in Coq - A Formally-Verified C Compiler Supporting Floating-Point Arithmetic - Verified Compilation of Floating-Point Computations - Innocuous Double Rounding of Basic Arithmetic Operations - Formal Proofs of Rounding Error Bounds - Computer Arithmetic and Formal Proofs
- URL: http://flocq.gforge.inria.fr/

6.8. Gappa

The Gappa tool for automated proofs of arithmetic properties
KEYWORDS: Floating-point - Arithmetic code - Software Verification - Constraint solving
FUNCTIONAL DESCRIPTION: Gappa is a tool intended to help formally verifying numerical programs dealing with floating-point or fixed-point arithmetic. It has been used to write robust floating-point filters for CGAL and it is used to verify elementary functions in CRlibm. While Gappa is intended to be used directly, it can also act as a backend prover for the Why3 software verification plateform or as an automatic tactic for the Coq proof assistant.

- Participant: Guillaume Melquiond
- Contact: Guillaume Melquiond
- Publications: Generating formally certified bounds on values and round-off errors - Formal certification of arithmetic filters for geometric predicates - Assisted verification of elementary functions - From interval arithmetic to program verification - Formally Certified Floating-Point Filters For Homogeneous Geometric Predicates - Combining Coq and Gappa for Certifying Floating-Point Programs - Handbook of Floating-Point Arithmetic - Certifying the floating-point implementation of an elementary function using Gappa - Automations for verifying floating-point algorithms - Automating the verification of floating-point algorithms - Computer Arithmetic and Formal Proofs
- URL: http://gappa.gforge.inria.fr/

6.9. Why3

The Why3 environment for deductive verification
KEYWORDS: Formal methods - Trusted software - Software Verification - Deductive program verification
FUNCTIONAL DESCRIPTION: Why3 is an environment for deductive program verification. It provides a rich language for specification and programming, called WhyML, and relies on external theorem provers, both automated and interactive, to discharge verification conditions. Why3 comes with a standard library of logical theories (integer and real arithmetic, Boolean operations, sets and maps, etc.) and basic programming data structures (arrays, queues, hash tables, etc.). A user can write WhyML programs directly and get correct-by-construction OCaml programs through an automated extraction mechanism. WhyML is also used as an intermediate language for the verification of C, Java, or Ada programs.

- Participants: Andriy Paskevych, Claude Marché, François Bobot, Guillaume Melquiond, Jean-Christophe Filliâtre, Levs Gondelmans and Martin Clochard
- Partners: CNRS - Université Paris-Sud
- Contact: Claude Marché
- URL: http://why3.lri.fr/

VERIDIS Project-Team

6. New Software and Platforms

6.1. Redlog

Reduce Logic System
KEYWORDS: Computer algebra system (CAS) - First-order logic - Constraint solving
SCIENTIFIC DESCRIPTION: Redlog is an integral part of the interactive computer algebra system Reduce. It supplements Reduce's comprehensive collection of powerful methods from symbolic computation by supplying more than 100 functions on first-order formulas.

Redlog generally works with interpreted first-order logic in contrast to free first-order logic. Each first-order formula in Redlog must exclusively contain atoms from one particular Redlog-supported theory, which corresponds to a choice of admissible functions and relations with fixed semantics. Redlog-supported theories include Nonlinear Real Arithmetic (Real Closed Fields), Presburger Arithmetic, Parametric QSAT, and many more.
NEWS OF THE YEAR: In 2017, there was a strong focus on applications of Redlog. With the final phase of the ANR-DFG Project SMArT, Redlog was integrated with the SMT solver veriT. That combination, as well as a stand-alone version of Redlog, participated in the SMT competition SMTCOMP 2017. All configurations performed very well, the stand-alone version won the category NRA (nonlinear real arithmetic).

On the scientific side, we made significant progress with the symbolic bifurcation analysis for biological networks.

Redlog technology for biological network analysis from last year, viz. subtropical solving, has raised considerable attention in the SMT community, where it has been adopted and triggered new research.

- Participant: Thomas Sturm
- Contact: Thomas Sturm
- URL: http://www.redlog.eu/

6.2. SPASS

KEYWORD: First-order logic
SCIENTIFIC DESCRIPTION: The classic SPASS is an automated theorem prover based on superposition that handles first-order logic with equality and several extensions for particular classes of theories. With version SPASS 3.9 we have stopped the development of the classic prover and have started the bottom-up development of SPASS 4.0 that will actually be a workbench of automated reasoning tools. Furthermore, we use SPASS 3.9 as a test bed for the development of new calculi.

Meanwhile we have released the second version of SPASS-IQ, our solver for linear integer arithmetic that we are currently extending to real and mixed real-integer arithmetic. We didn't release SPASS-SATT yet, instead we further investigated the use of redundency elimination in SAT solving and underlying implementation techniques. Our aim is a new approach to SAT solving that needs fewer conflicts (on average) *and* is faster than the current state-of-the art solvers. Furthermore, we have developed a new calculus and first prototypical implementation of a SAT solver with mixed OR/XOR clauses.

SPASS 3.9 has been used as the basis for SPASS-AR, an new approximation refinement theorem proving approach.
FUNCTIONAL DESCRIPTION: SPASS is an automated theorem prover based on superposition that handles first-order logic with equality and several extensions for particular classes of theories.

- Contact: Christoph Weidenbach
- URL: http://www.spass-prover.org/

6.3. TLAPS

TLA+ proof system
KEYWORD: Proof assistant
FUNCTIONAL DESCRIPTION: TLAPS is a platform for developing and mechanically verifying proofs about TLA+ specifications. The TLA+ proof language is hierarchical and explicit, allowing a user to decompose the overall proof into proof steps that can be checked independently. TLAPS consists of a proof manager that interprets the proof language and generates a collection of proof obligations that are sent to backend verifiers. The current backends include the tableau-based prover Zenon for first-order logic, Isabelle/TLA+, an encoding of TLA+ set theory as an object logic in the logical framework Isabelle, an SMT backend designed for use with any SMT-lib compatible solver, and an interface to a decision procedure for propositional temporal logic.
NEWS OF THE YEAR: In 2017, we have continued to work on a complete reimplementation of the proof manager. Its objectives are a cleaner interaction with the TLA$^+$ front-ends, in particular SANY, the standard parser and semantic analyzer. The reimplementation is also necessary for extending the scope of the fragment of TLA$^+$ that is handled by TLAPS, in particular full temporal logic and module instantiation.

- Participants: Damien Doligez, Stephan Merz and Martin Riener
- Contact: Stephan Merz
- URL: https://tla.msr-inria.inria.fr/tlaps/content/Home.html

6.4. veriT

KEYWORDS: Automated deduction - Formula solving - Verification
SCIENTIFIC DESCRIPTION: veriT comprises a SAT solver, a decision procedure for uninterpreted symbols based on congruence closure, a simplex-based decision procedure for linear arithmetic, and instantiation-based quantifier handling.
FUNCTIONAL DESCRIPTION: VeriT is an open, trustable and efficient SMT (Satisfiability Modulo Theories) solver, featuring efficient decision procedure for uninterpreted symbols and linear arithmetic, and quantifier reasoning.
NEWS OF THE YEAR: Efforts in 2017 have been focused on non-linear arithmetic reasoning and quantifier handling. The reasoning capabilities of veriT have been significantly improved along those two axes.

The veriT solver participated in the SMT competition SMT-COMP 2017 with good results.

We target applications where validation of formulas is crucial, such as the validation of TLA$^+$ and B specifications, and work together with the developers of the respective verification platforms to make veriT even more useful in practice. The solver is available as a plugin for the Rodin platform, it is integrated within the Atelier B.

- Participants: Haniel Barbosa, Daniel El Ouraoui, Pascal Fontaine and Hans-Jörg Schurr
- Partner: Université de Lorraine
- Contact: Pascal Fontaine
- URL: http://www.veriT-solver.org

6.5. Nunchaku

The Nunchaku Higher-Order Model Finder
KEYWORDS: Proof - Higher-order logic
SCIENTIFIC DESCRIPTION: Nunchaku is a model finder for higher-order logic, with dedicated support for various definitional principles. It is designed to work as a backend for various proof assistants (notably Isabelle/HOL and Coq) and to use state-of-the-art model finders and other solvers as backends.
FUNCTIONAL DESCRIPTION: Nunchaku is a model finder (counterexample generator) for higher-order logic.
NEWS OF THE YEAR: A noteworthy development this year is the creation of a backend called SMBC, based on new ideas by Cruanes about how to combine SAT solving and narrowing.

- Participants: Jasmin Christian Blanchette and Simon Cruanes
- Contact: Jasmin Christian Blanchette
- URL: https://github.com/nunchaku-inria

CARTE Team

6. New Software and Platforms

6.1. Software

6.1.1. FiatLux

FiatLux is a simulation program for cellular automata developed by Nazim Fatès. The project is currently available at the Inria GForge. It is under the CeCILL license.

CIDRE Project-Team

6. New Software and Platforms

6.1. Blare

To detect intrusion using information flows
KEYWORDS: Cybersecurity - Intrusion Detection Systems (IDS) - Data Leakage Protection
SCIENTIFIC DESCRIPTION: Blare implements our approach of illegal information flow detection for a single node (Android and Linux kernel, JVM) and a set of nodes (monitoring of flows between linux machines).
FUNCTIONAL DESCRIPTION: Blare IDS is a set of tools that implements our approach to illegal information flow detection for a single node and a set of nodes.
NEWS OF THE YEAR: During this year, Laurent Georget has modified the implementation of Blare in order to correctly monitor the kernel system calls with LSM hooks. He add also ported this new version of Blare to the Lollipop Android emulator.

- Partner: CentraleSupélec
- Contact: Frédéric Tronel
- Publications: Information Flow Tracking for Linux Handling Concurrent System Calls and Shared Memory - Verifying the Reliability of Operating System-Level Information Flow Control Systems in Linux - Monitoring both OS and program level information flows to detect intrusions against network servers - Experimenting a Policy-Based HIDS Based on an Information Flow Control Model - Introducing reference flow control for intrusion detection at the OS level - Blare Tools: A Policy-Based Intrusion Detection System Automatically Set by the Security Policy - Diagnosing intrusions in Android operating system using system flow graph - Intrusion detection in distributed systems, an approach based on taint marking - BSPL: A Language to Specify and Compose Fine-grained Information Flow Policies - Information Flow Policies vs Malware - A taint marking approach to confidentiality violation detection - Designing information flow policies for Android's operating system - Information Flow Control for Intrusion Detection derived from MAC Policy - Flow based interpretation of access control: Detection of illegal information flows - A taint marking approach to confidentiality violation detection
- URL: http://www.blare-ids.org/

6.2. GNG

Security Supervision by Alert Correlation
KEYWORDS: Intrusion Detection Systems (IDS) - SIEM
SCIENTIFIC DESCRIPTION: GNG is an intrusion detection system that correlates different sources (such as different logs) in order to identify attacks against the system. The attack scenarios are defined using the Attack Description Langage (ADeLe) proposed by our team, and are internally translated to attack recognition automatons. GNG intends to define time efficient algorithms based on these automatons to recognize complex attack scenarios.

- Partner: CentraleSupélec
- Contact: Eric Totel
- Publication: A Language Driven Intrusion Detection System for Events and Alerts Correlation
- URL: http://www.rennes.supelec.fr/ren/perso/etotel/GNG/index.html

6.3. GroddDroid

KEYWORDS: Android - Detection - Malware

SCIENTIFIC DESCRIPTION: GroddDroid automates the dynamic analysis of a malware. When a piece of suspicious code is detected, groddDroid interacts with the user interface and eventually forces the execution of the identified code. Using Blare (Information Flow Monitor), GroddDroid monitors how an execution contaminates the operating system. The output of GroddDroid can be visualized in an web browser. GroddDroid is used by the Kharon software.
FUNCTIONAL DESCRIPTION: GroddDroid 1 - locates suspicious code in Android application 2 - computes execution paths towards suspicious code 3 - forces executions of suspicious code 4 - automate the execution of a malware or a regular Android application
NEWS OF THE YEAR: In 2017, GroddDroid has integrated the work of Mourad Leslous, who have implemented GFinder. GPFinder improves the computation of control flow paths by taking into account the Android framework. The end of the year has been used to clean the code and to improves the graphical interface.

- Partners: CentraleSupélec - Insa Centre Val-de-Loire
- Contact: Valérie Viet Triem Tong
- Publications: Kharon dataset: Android malware under a microscope - GroddDroid: a Gorilla for Triggering Malicious Behaviors - GPFinder: Tracking the Invisible in Android Malware - Information flows at OS level unmask sophisticated Android malware
- URL: http://kharon.gforge.inria.fr/grodddroid.html

6.4. Kharon

KEYWORDS: Android - Malware - Dynamic Analysis
FUNCTIONAL DESCRIPTION: Kharon is a software for managing Android application analysis. Kharon uses the results of the GroddDroid software. The user can submit one or several applications to Kharon and get a graph of the innformation flows that occured at system level and that have been caused by the application.

Kharon is used in the Kharon platform for the analysis of malicious applications. This platform is deployed at the high security laboratory (LHS) of Rennes.

- Author: Sébastien Campion
- Partners: CentraleSupélec - Insa Centre Val-de-Loire
- Contact: Valérie Viet Triem Tong
- URL: http://kharon.gforge.inria.fr/

6.5. StarLord

KEYWORDS: Security - SIEM
FUNCTIONAL DESCRIPTION: In the domain of security event visualisation, we have developed a prototype called StarLord. Basically, this software is able to parse heterogeneous logs, and to extract from each line of logs a set of security objects. Moreover, some of these objects appears in several lines of different logs. These lines are thus linked by the sharing of one or more security objects. When we analyse the lines of logs, we are thus able to generate graphs that represents the links between the different objects discovered in the logs. These graphs are thus displayed in 3D in order for the administrator to investigate easily the relations between the logs and the relations between the logs and some particular indicators of compromission. The tool permits to discover visually the activity of an attacker on the supervised system.

- Authors: Ludovic Mé, Eric Totel, Nicolas Prigent and Laetitia Leichtnam
- Contact: Eric Totel
- Publication: STARLORD: Linked Security Data Exploration in a 3D Graph

6.6. SpecCert

KEYWORDS: Formal methods - Coq

FUNCTIONAL DESCRIPTION: SpecCert is a framework for specifying and verifying Hardware-based Security Enforcement (HSE) mechanisms against hardware architecture models. HSE mechanisms form a class of security enforcement mechanism such that a set of trusted software components relies on hardware functions to enforce a security policy.

- Participant: Thomas Letan
- Partners: ANSSI - CentraleSupélec
- Contact: Guillaume Hiet
- Publications: SpecCert: Specifying and Verifying Hardware-based Security Enforcement - SpecCert: Specifying and Verifying Hardware-based Software Enforcement
- URL: https://github.com/lethom/speccert

6.7. HardBlare

KEYWORDS: Intrusion Detection Systems (IDS) - FPGA - Static analysis
FUNCTIONAL DESCRIPTION: HardBlare is a hardware/software framework to implement hardware DIFC on Xilinx Zynq Platform. HardBlare consists of three components : 1) the VHDL code of the coprocessor, 2) a modified LLVM compiler to compute the static analysis, and 3) a dedicated Linux kernel. This last component is a specific version of the Blare monitor.

- Partners: CentraleSupélec - Lab-STICC
- Contact: Guillaume Hiet
- Publications: ARMHEx: A hardware extension for DIFT on ARM-based SoCs - ARMHEx: a framework for efficient DIFT in real-world SoCs - ARMHEx: embedded security through hardware-enhanced information flow tracking - HardBlare: a Hardware-Assisted Approach for Dynamic Information Flow Tracking - A portable approach for SoC-based Dynamic Information Flow Tracking implementations - Towards a hardware-assisted information flow tracking ecosystem for ARM processors - HardBlare: an efficient hardware-assisted DIFC for non-modified embedded processors

6.8. Conductor

KEYWORDS: Intrusion Detection Systems (IDS) - Static analysis - Instrumentation
FUNCTIONAL DESCRIPTION: Conductor contains three main components: a static analysis to extract the expected behavior of the target, an instrumentation module to add instructions to the target's code in order to send messages to the co-processor, and an intrusion detection engine executed on the co-processor. The latter processes the messages sent by the instrumented target, describing its current behavior. This behavior is then compared against the expected behavior previously extracted by the static analysis.

- Participants: Ronny Chevalier, Guillaume Hiet, Maugan Villatel and David Plaquin
- Partners: CentraleSupélec - HP Labs
- Contact: Ronny Chevalier
- Publication: Co-processor-based Behavior Monitoring: Application to the Detection of Attacks Against the System Management Mode

6.9. Platforms

6.9.1. Kharon platform

The Kharon platform is under development in the LHS of Rennes and should be ready to use in the beginning of 2018. This experimental platform aims to analyze Android malware using a set of software developed by the CIDRE team. Software that are involved are:

- The Blare IDS http://www.blare-ids.org/, and in particular the AndroBlare version, for tracking information flows of malware;
- The GroddDroid software http://kharon.gforge.inria.fr/grodddroid.html, for manipulating the malware statically and dynamically;
- The GPFinder software http://kharon.gforge.inria.fr/gpfinder.html, for computing paths in the malware's control flow;
- The kharon software that handles the orchestration of a bunch of malware, the server and a set of smartphones.

The Kharon platform will be used for analysing malware as soon as they appear in the wild. The analysis results will be stored for further experiments and statistics.

COMETE Project-Team

5. New Software and Platforms

5.1. Location Guard

KEYWORDS: Privacy - Geolocation - Browser Extensions
SCIENTIFIC DESCRIPTION: The purpose of Location Guard is to implement obfuscation techniques for achieving location privacy, in a an easy and intuitive way that makes them available to the general public. Various modern applications, running either on smartphones or on the web, allow third parties to obtain the user's location. A smartphone application can obtain this information from the operating system using a system call, while web application obtain it from the browser using a JavaScript call.
FUNCTIONAL DESCRIPTION: Websites can ask the browser for your location (via JavaScript). When they do so, the browser first asks your permission, and if you accept, it detects your location (typically by transmitting a list of available wifi access points to a geolocation provider such as Google Location Services, or via GPS if available) and gives it to the website.

Location Guard is a browser extension that intercepts this procedure. The permission dialog appears as usual, and you can still choose to deny. If you give permission, then Location Guard obtains your location and adds "random noise" to it, creating a fake location. Only the fake location is then given to the website.

In 2017 there was a major update to the Firefox version of Location Guard, to make it compatible with the Firefox Quantum. This latest Firefox version discontinued support for the legacy addon API, so Location Guard had to be adapted to the new WebExtensions API.

Moreover, the latest version implements new features requested by users, such as the ability to search for a fixed location, as well as bugfixes.

- Participants: Catuscia Palamidessi, Konstantinos Chatzikokolakis, Marco Stronati, Miguel Andrés and Nicolas Bordenabe
- Contact: Konstantinos Chatzikokolakis
- URL: https://github.com/chatziko/location-guard

5.2. libqif - A Quantitative Information Flow C++ Toolkit Library

KEYWORDS: Information leakage - Privacy - C++ - Linear optimization
FUNCTIONAL DESCRIPTION: The goal of libqif is to provide an efficient C++ toolkit implementing a variety of techniques and algorithms from the area of quantitative information flow and differential privacy. We plan to implement all techniques produced by Comète in recent years, as well as several ones produced outside the group, giving the ability to privacy researchers to reproduce our results and compare different techniques in a uniform and efficient framework.

Some of these techniques were previously implemented in an ad-hoc fashion, in small, incompatible with each-other, non-maintained and usually inefficient tools, used only for the purposes of a single paper and then abandoned. We aim at reimplementing those – as well as adding several new ones not previously implemented – in a structured, efficient and maintainable manner, providing a tool of great value for future research. Of particular interest is the ability to easily re-run evaluations, experiments and case-studies from all our papers, which will be of great value for comparing new research results in the future.

The library's development continued in 2017 with several new added features. The project's git repository shows for this year 33 commits by 2 contributors. The new functionality was directly applied to the exeperimental results of several publications of the team (PETS'17, GameSec'17, VALUETOOLS'17).

- Contact: Konstantinos Chatzikokolakis
- URL: https://github.com/chatziko/libqif

5.3. dspacenet

Distributed-Spaces Network.
KEYWORDS: Social networks - Distributed programming
FUNCTIONAL DESCRIPTION: DSpaceNet is a tool for social networking based on multi-agent spatial and timed concurrent constraint language.

I - The fundamental structure of DSPaceNet is that of *space*: A space may contain

(1) spatial-mobile-reactive tcc programs, and (2) other spaces.

Furthermore, (3) each space belongs to a given agent. Thus, a space of an agent j within the space of agent i means that agent i allows agent j to use a computation sub-space within its space.

II - The fundamental operation of DSPaceNet is that of *program posting*: In each time unit, agents can post spatial-mobile-reactive tcc programs in the spaces they are allowed to do so (ordinary message posting corresponds to the posting of tell processes). Thus, an agent can for example post a watchdog tcc process to react to messages in their space, e.g. whenever (*happy b*frank*) do tell("thank you!"). More complex mobile programs are also allowed (see below).

The language of programs is a spatial mobile extension of tcc programs:

$P, Q... := tell(c)|whencdoP||nextP|P||Q|unlesscnextP|[P]\_i| \uparrow \_iP|recX.P$

computation of timed processes proceeds as in tcc. The spatial construct [P]_i runs P in the space of agent i and the mobile process uparrow_i P, extrudes P from the space of i. By combining space and mobility, arbitrary processes can be moved from one a space into another. For example, one could send a trojan watchdog to another space for spying for a given message and report back to one's space.

III- Constraint systems can be used to specify advance text message deduction, arithmetic deductions, scheduling, etc.

IV - Epistemic Interpretation of spaces can be used to derive whether they are users with conflicting/inconsistent information, or whether a group of agents may be able to deduce certain message.

V - The scheduling of agent requests for program posts, privacy settings, friendship lists are handled by an external interface. For example, one could use type systems to check whether a program complies with privacy settings (for example checking that the a program does not move other program into a space it is not allowed into).

- Partner: Pontificia Universidad Javeriana Cali
- Contact: Frank Valencia
- URL: http://www.dspacenet.com

DATASPHERE Team

6. New Software and Platforms

6.1. Platforms

The team participated to the development of the following software platforms.

6.1.1. DNS data analysis

Data analytics tools for DNS data analysis were developed in a cooperation with ICT, Chinese Academy of Sciences in the frame of the thesis of Jingxiu SU [9].

6.1.2. Advokat

Distributed aggregation mechanisms preserving confidentiality for application such as online voting were developed in the frame of the thesis of Robert Riemann [11].

6.1.3. BGP Geopolitics

An observatory of global BGP connectivity has been developed that is used to monitor in real time BGP level attacks. In addition, a set of tools were developed to analyse the structure of information propagation over social networks.

PESTO Project-Team

6. New Software and Platforms

6.1. Akiss

AKISS: Active Knowledge in Security Protocols
KEYWORDS: Security - Verification
FUNCTIONAL DESCRIPTION: Akiss (Active Knowledge in Security Protocols) is a tool for verifying indistinguishability properties in cryptographic protocols, modelled as trace equivalence in a process calculus. Indistinguishability is used to model a variety of properties including anonymity properties, strong versions of confidentiality and resistance against offline guessing attacks, etc. Akiss implements a procedure to verify equivalence properties for a bounded number of sessions based on a fully abstract modelling of the traces of a bounded number of sessions of the protocols into first-order Horn clauses and a dedicated resolution procedure. The procedure can handle a large set of cryptographic primitives, namely those that can be modeled by an optimally reducing convergent rewrite system.

- Contact: Steve Kremer
- URL: https://github.com/akiss

6.2. Belenios

Belenios - Verifiable online voting system
KEYWORD: E-voting
FUNCTIONAL DESCRIPTION: Belenios is an online voting system that provides confidentiality and verifiability. End-to-end verifiablity relies on the fact that the ballot box is public (voters can check that their ballots have been received) and on the fact that the tally is publicly verifiable (anyone can recount the votes). Confidentiality relies on the encryption of the votes and the distribution of the decryption key.

Belenios builds upon Helios, a voting protocol used in several elections. The main design enhancement of Belenios vs Helios is that the ballot box can no longer add (fake) ballots, due to the use of credentials.

- Participants: Pierrick Gaudry, Stéphane Glondu and Véronique Cortier
- Partners: CNRS - Inria
- Contact: Stéphane Glondu
- URL: http://belenios.gforge.inria.fr/

6.3. CL-AtSe

Constraint Logic based Attack Searcher
KEYWORDS: Security - Verification - Web Services
FUNCTIONAL DESCRIPTION: CL-AtSe is a Constraint Logic based Attack Searcher for security protocols and services. The main idea in CL-AtSe consists in running the protocol or set of services in all possible ways by representing families of traces with positive or negative constraints on the intruder knowledge, on variable values, on sets, etc. Thus, each run of a service step consists in adding new constraints on the current intruder and environment state, reducing these constraints down to a normalized form for which satisfiability is easily decidable, and decide whether some security property has been violated up to this point.

- Participants: Mathieu Turuani and Tigran Avanesov
- Contact: Mathieu Turuani
- URL: https://cassis.loria.fr/wiki/Wiki.jsp?page=Cl-Atse

6.4. Deepsec

DEciding Equivalence Properties in SECurity protocols
KEYWORDS: Security - Verification
FUNCTIONAL DESCRIPTION: DeepSec (DEciding Equivalence Properties in SECurity protocols) is a tool for verifying indistinguishability properties in cryptographic protocols, modelled as trace equivalence in a process calculus. Indistinguishability is used to model a variety of properties including anonymity properties, strong versions of confidentiality and resistance against offline guessing attacks, etc. DeepSec implements a decision procedure to verify trace equivalence for a bounded number of sessions and cryptographic primitives modeled by a subterm convergent destructor rewrite system. The procedure is based on constraint solving techniques. Several new features are currently being developed including the possibility to verify labelled bisimilarity and session equivalence. Optimizations to improve efficiency and interface improvements are also under development.

- Contact: Vincent Cheval
- URL: https://github.com/DeepSec-prover/deepsec

6.5. Tamarin

TAMARIN prover
KEYWORDS: Security - Verification
FUNCTIONAL DESCRIPTION: The TAMARIN prover is a security protocol verification tool that supports both falsification and unbounded verification of security protocols specified as multiset rewriting systems with respect to (temporal) first-order properties and a message theory that models Diffie-Hellman exponentiation, bilinear pairing, multisets, and exclusive-or (XOR), combined with a user-defined convergent rewriting theory. Its main advantages are its ability to handle stateful protocols and its interactive proof mode. Moreover, it has recently been extended to verify equivalence properties. The tool is developed jointly by the PESTO team, the Institute of Information Security at ETH Zurich, and the University of Oxford. In a joint effort, the partners wrote and published a user manual in 2016, available from the Tamarin website.

- Contact: Jannik Dreier
- URL: http://tamarin-prover.github.io/

6.6. SAPIC

SAPIC: Stateful Applied Pi Calculus
KEYWORDS: Security - Verification
FUNCTIONAL DESCRIPTION: SAPIC is a tool that translates protocols from a high-level protocol description language akin to the applied pi-calculus into multiset rewrite rules, that can then be analysed using the TAMARIN prover. TAMARIN has also been extended with dedicated heuristics that exploit the form of translated rules and favor termination.

SAPIC offers support for the analysis of protocols that include states, for example Hardware Security Tokens communicating with a possibly malicious user, or protocols that rely on databases. It also allows us to verify liveness properties and a recent extension adds a notion of location and reporting used for modelling trusted execution environments. It has been successfully applied on several case studies including the Yubikey authentication protocol, and extensions of the PKCS#11 standard. SAPIC also includes support for verifying liveness properties, which are for instance important in fair exchange and contract signing protocols, as well as support for constructions useful when modelling isolated execution environments.

SAPIC has been integrated as a plugin in TAMARIN and is now part of the TAMARIN distribution.

- Contact: Steve Kremer
- URL: http://sapic.gforge.inria.fr/

6.7. TypeEquiv

A type checker for privacy properties
KEYWORDS: Security - Cryptographic protocol - Privacy
FUNCTIONAL DESCRIPTION: TypeEquiv takes as input the specification of a pair of security protocols, written in a dialect of the applied-pi calculus, together with some type annotations. It checks whether the two protocols are in equivalence or not.

- Partner: Technische Universität Wien
- Contact: Véronique Cortier

PRIVATICS Project-Team

5. New Software and Platforms

5.1. FECFRAME

FEC Framework following RFC 6363 specifications (https://datatracker.ietf.org/doc/rfc6363/)
KEYWORDS: Error Correction Code - Content delivery protocol - Robust transmission
FUNCTIONAL DESCRIPTION: This sofware implements the FECFRAME IETF standard (RFC 6363) co-authored by V. Roca, and is compliant with 3GPP specifications for mobile terminals. It enables the simultaneous transmission of multimedia flows to one or several destinations, while being robust to packet erasures that happen on wireless networks (e.g., 4G or Wifi). This software relies on the OpenFEC library (the open-source http://openfec.org version or the commercial version) that provides the erasure correction codes (or FEC) and thereby offer robustness in front of packet erasures.

- Participant: Vincent Roca
- Contact: Vincent Roca

5.2. Mobilitcs

FUNCTIONAL DESCRIPTION: Mobilitics is a joint project, started in 2012 between Inria and CNIL, which targets privacy issues on smartphones. The goal is to analyze the behavior of smartphones applications and their operating system regarding users private data, that is, the time they are accessed or sent to third party companies usually neither with user's awareness nor consent.

In the presence of a wide range of different smartphones available in terms of operating systems and hardware architecture, Mobilitics project focuses actually its study on the two mostly used mobile platforms, IOS (Iphone) and Android. Both versions of the Mobilitics software: (1) capture any access to private data, any modification (e.g., ciphering or hashing of private data), or transmission of data to remote locations on the Internet, (2) store these events in a local database on the phone for offline analysis, and (3) provide the ability to perform an in depth database analysis in order to identify personnal information leakage.

- Authors: Jagdish Achara, James-Douglass Lefruit, Claude Castelluccia, Franck Baudot, Geoffrey Delcroix, Gwendal Le Grand, Stéphane Petitcolas and Vincent Roca
- Contact: Claude Castelluccia

5.3. MyTrackingChoices

KEYWORDS: Privacy - User control
FUNCTIONAL DESCRIPTION: This extension lets you control how you are being tracked on the Internet. It allows you to choose the categories (e.g., health, adult) of the websites where you don't want to be tracked on. When you browse the web, your visited webpages will be categorized on the fly and, depending on your choices, the extension will block the trackers (webpage by webpage) or not.

Existing anti-tracking (Ghostery, Disconnect etc.) and ad-blocking (AdBlock Plus etc.) tools block almost ALL trackers and as a result, ads. This has a negative impact on the Internet economy because free services/content on the Internet are fuelled by ads. As a result, websites are starting to block access to their content if they detect use of Ad-blockers or they ask users to move to a subscription-based model (where users have to pay to get access to the website).

This extension is testing another approach: It is trying to find a trade-off between privacy and economy, that would allow users to protect their privacy while still accessing to free content.

It is based on the assumption that most people are not against advertisements, but want to keep control over their data. We believe that some sites are more sensitive than others. In fact, most people don't want to be tracked on "sensitive" websites (for example related to religion, health,...), but don't see any problem to be tracked on less sensitive ones (such as news, sport,...). This extension allows you to take control and specify which on which categories of sites you don't want to be tracked on! Furthermore, the extension also gives you the option to block the trackers on specific websites.

- Contact: Claude Castelluccia
- URL: https://addons.mozilla.org/FR/firefox/addon/mytrackingchoices/

5.4. OMEN+

FUNCTIONAL DESCRIPTION: Omen+ is a password cracker following our previous work. It is used to guess possible passwords based on specific information about the target. It can also be used to check the strength of user password by effectively looking at the similarity of that password with both usual structures and information relative to the user, such as his name, birth date...

It is based on a Markov analysis of known passwords to build guesses. The previous work Omen needs to be cleaned in order to be scaled to real problems and to be distributed or transfered to the security community (maintainability): eventually it will become an open source software. The main challenge of Omen+ is to optimize the memory consumption.

- Participants: Claude Castelluccia and Pierre Rouveyrol
- Contact: Claude Castelluccia

5.5. OPENFEC

KEYWORD: Error Correction Code
FUNCTIONAL DESCRIPTION: OpenFEC is a C-language implementation of several Application-Level Forward Erasure Correction (AL-FEC) codecs, namely: Reed-Solomon (RFC 5510), LDPC-Staircase (RFC 5170) codes, and RLC (https://datatracker.ietf.org/doc/draft-ietf-tsvwg-rlc-fec-scheme/). Two versions are available: an open-source, unsupported version (http://openfec.org), and an advanced version commercialized by the Expway SME.

RELEASE FUNCTIONAL DESCRIPTION: Added support of RLC codes (Random Linear Codes), based on a sliding encoding window.

- Participants: Christophe Neumann, Belkacem Teibi, Jérome Lacan, Jonathan Detchart, Julien Laboure, Kevin Chaumont, Mathieu Cunche and Vincent Roca
- Partner: Expway
- Contact: Vincent Roca
- URL: http://openfec.org/

PROSECCO Project-Team

6. New Software and Platforms

6.1. Cryptosense Analyzer

SCIENTIFIC DESCRIPTION: Cryptosense Analyzer (formerly known as Tookan) is a security analysis tool for cryptographic devices such as smartcards, security tokens and Hardware Security Modules that support the most widely-used industry standard interface, RSA PKCS#11. Each device implements PKCS#11 in a slightly different way since the standard is quite open, but finding a subset of the standard that results in a secure device, i.e. one where cryptographic keys cannot be revealed in clear, is actually rather tricky. Cryptosense Analyzer analyses a device by first reverse engineering the exact implementation of PKCS#11 in use, then building a logical model of this implementation for a model checker, calling a model checker to search for attacks, and in the case where an attack is found, executing it directly on the device. It has been used to find at least a dozen previously unknown flaws in commercially available devices.
FUNCTIONAL DESCRIPTION: Cryptosense Analyzer (formerly known as Tookan) is a security analysis tool for cryptographic devices such as smartcards,

- Participants: Graham Steel and Romain Bardou
- Contact: Graham Steel
- URL: https://cryptosense.com/

6.2. CryptoVerif

Cryptographic protocol verifier in the computational model
KEYWORDS: Security - Verification - Cryptographic protocol
FUNCTIONAL DESCRIPTION: CryptoVerif 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 and correspondences, which include in particular authentication. It 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, and Diffie-Hellman key agreements. It also provides an explicit formula that gives the probability of breaking the protocol as a function of the probability of breaking each primitives, this is the exact security framework.
NEWS OF THE YEAR: We made several case studies using CryptoVerif (Signal, TLS 1.3 Draft 18, ARINC 823 avionic protocol) and have made a few technical improvements.

- Participants: Bruno Blanchet and David Cadé
- Contact: Bruno Blanchet
- Publications: Proved Implementations of Cryptographic Protocols in the Computational Model - Proved Generation of Implementations from Computationally Secure Protocol Specifications - Verified Models and Reference Implementations for the TLS 1.3 Standard Candidate - Verified Models and Reference Implementations for the TLS 1.3 Standard Candidate - Symbolic and Computational Mechanized Verification of the ARINC823 Avionic Protocols - Automated Verification for Secure Messaging Protocols and Their Implementations: A Symbolic and Computational Approach
- URL: http://cryptoverif.inria.fr/

6.3. F*

FStar
KEYWORDS: Programming language - Software Verification

FUNCTIONAL DESCRIPTION: F* is a new higher order, effectful programming language (like ML) designed with program verification in mind. Its type system is based on a core that resembles System Fw (hence the name), but is extended with dependent types, refined monadic effects, refinement types, and higher kinds. Together, these features allow expressing precise and compact specifications for programs, including functional correctness properties. The F* type-checker aims to prove that programs meet their specifications using an automated theorem prover (usually Z3) behind the scenes to discharge proof obligations. Programs written in F* can be translated to OCaml, F#, or JavaScript for execution.

- Participants: Antoine Delignat-Lavaud, Catalin Hritcu, Cédric Fournet, Chantal Keller, Karthikeyan Bhargavan and Pierre-Yves Strub
- Contact: Catalin Hritcu
- URL: https://www.fstar-lang.org/

6.4. miTLS

KEYWORDS: Cryptographic protocol - Software Verification
FUNCTIONAL DESCRIPTION: miTLS is a verified reference implementation of the TLS protocol. Our code fully supports its wire formats, ciphersuites, sessions and connections, re-handshakes and resumptions, alerts and errors, and data fragmentation, as prescribed in the RFCs, it interoperates with mainstream web browsers and servers. At the same time, our code is carefully structured to enable its modular, automated verification, from its main API down to computational assumptions on its cryptographic algorithms.

- Participants: Alfredo Pironti, Antoine Delignat-Lavaud, Cédric Fournet, Jean-Karim Zinzindohoué, Karthikeyan Bhargavan, Pierre-Yves Strub and Santiago Zanella-Béguelin
- Contact: Karthikeyan Bhargavan
- URL: https://github.com/mitls/mitls-fstar

6.5. ProVerif

KEYWORDS: Security - Verification - Cryptographic protocol
FUNCTIONAL DESCRIPTION: ProVerif is an automatic security protocol verifier in the symbolic 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 verify various security properties (secrecy, authentication, process equivalences).

It can handle many different cryptographic primitives, specified 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.
NEWS OF THE YEAR: Marc Sylvestre improved the display of attacks, in particular by showing the computations performed by the attacker to obtain the messages sent in the attack, and by explaining why the found trace breaks the considered security property. He also developed an interactive simulator that allows the user to run the protocol step by step. We also made several case studies using this tool (Signal, TLS 1.3 Draft 18, ARINC 823 avionic protocol).

- Participants: Bruno Blanchet, Marc Sylvestre and Vincent Cheval
- Contact: Bruno Blanchet
- Publications: Automated Reasoning for Equivalences in the Applied Pi Calculus with Barriers - Automated reasoning for equivalences in the applied pi calculus with barriers - Modeling and Verifying Security Protocols with the Applied Pi Calculus and ProVerif - Automatic Verification of Security Protocols in the Symbolic Model: The Verifier ProVerif - Verified Models and Reference Implementations for the TLS 1.3 Standard Candidate - Verified Models and Reference Implementations for the TLS 1.3 Standard Candidate - Automated Verification for Secure Messaging Protocols and Their Implementations: A Symbolic and Computational Approach - Symbolic and Computational Mechanized Verification of the ARINC823 Avionic Protocols - Symbolic and Computational Mechanized Verification of the ARINC823 Avionic Protocols
- URL: http://proverif.inria.fr/

6.6. HACL*

High Assurance Cryptography Library
KEYWORDS: Cryptography - Software Verification
FUNCTIONAL DESCRIPTION: HACL* is a formally verified cryptographic library in F*, developed by the Prosecco team at Inria Paris in collaboration with Microsoft Research, as part of Project Everest.

HACL stands for High-Assurance Cryptographic Library and its design is inspired by discussions at the HACS series of workshops. The goal of this library is to develop verified C reference implementations for popular cryptographic primitives and to verify them for memory safety, functional correctness, and secret independence.

- Contact: Karthikeyan Bhargavan
- URL: https://github.com/mitls/hacl-star

TAMIS Team

6. New Software and Platforms

6.1. GNUnet

SCIENTIFIC DESCRIPTION: The GNUnet project seeks to answer the question what a modern Internet architecture should look like for a society that care about security and privacy. We are considering all layers of the existing well-known Internet, but are also providing new and higher-level abstractions (such as voting protocols, Byzantine consensus, etc.) that are today solved in application-specific ways. Research questions include the desired functionality of the overall stack, protocol design for the various layers as well as implementation considerations, i.e. how to implement the design securely.
FUNCTIONAL DESCRIPTION: GNUnet is a framework for secure peer-to-peer networking that does not use any centralized or otherwise trusted services. Our high-level goal is to provide a strong free software foundation for a global network that provides security and in particular respects privacy.

GNUnet started with an idea for anonymous censorship-resistant file-sharing, but has grown to incorporate other applications as well as many generic building blocks for secure networking applications. In particular, GNUnet now includes the GNU Name System, a privacy-preserving, decentralized public key infrastructure.

- Participants: Alvaro Garcia Recuero, Florian Dold, Gabor Toth, Hans Grothoff, Jeffrey Paul Burdges and Sree Hrsha Totakura
- Partner: The GNU Project
- Contact: Hans Grothoff
- URL: https://gnunet.org/

6.2. MHD

GNU libmicrohttpd
KEYWORDS: Embedded - Web 2.0
SCIENTIFIC DESCRIPTION: We are providing a standards compliant and complete implementation of the HTTP server protocol that allows developers to easily write correct HTTP servers. Key challenges include code size minimization (for IoT devices), performance (zero copy, scalability to 100k concurrent connections), portability and security. MHD is already widely used in production by both academic and industrial users. Ongoing research challenges include formal verification.
FUNCTIONAL DESCRIPTION: GNU libmicrohttpd is a small C library that is supposed to make it easy to run an HTTP server as part of another application.

- Participants: Evgeny Grin, Hans Grothoff and Sree Hrsha Totakura
- Partner: The GNU Project
- Contact: Hans Grothoff
- URL: http://www.gnu.org/software/libmicrohttpd/

6.3. PLASMA Lab

KEYWORDS: Energy - Statistics - Security - Runtime Analysis - Model Checker - Statistical - Model Checking - Aeronautics - Distributed systems
SCIENTIFIC DESCRIPTION: Statistical model checking (SMC) is a fast emerging technology for industrial scale verification and optimisation problems. SMC only requires an executable semantics and is not constrained by decidability. Therefore we can easily apply it to different modelling languages and logics. We have implemented in PLASMA Lab several advanced SMC algorithms that combine formal methods with statistical tests, which include techniques for rare events estimation and non-deterministic models.

FUNCTIONAL DESCRIPTION: PLASMA Lab is a compact, efficient and flexible platform for statistical model checking of stochastic models. PLASMA Lab includes simulators for PRISM models (Reactives Modules Language-RML) and Biological models. It also provides plugins that interface external simulators in order to support Matlab/Simulink, SytemC and LLVM . PLASMA Lab can be extended with new plugins to support other external simulators, and PLASMA Lab API can be used to embed the tool in other softwares. PLASMA Lab provide fast SMC algorithms, including advanced techniques for rare events simulation and nondeterministic models. These algorithms are designed in a distributed architecture to run large number of simulations on several computers, either on a local area network or grid. PLASMA Lab is implemented in Java with efficient data structures and low memory consumption
NEWS OF THE YEAR: In 2017 we have extended PLASMA Lab with a new simulator plugin that allows to verify LLVM code.

- Participants: Axel Legay, Jean Quilbeuf, Benoît Boyer, Kevin Corre, Louis-Marie Traonouez, Matthieu Simonin and Sean Sedwards
- Contact: Axel Legay
- URL: https://project.inria.fr/plasma-lab/

6.4. Taler

GNU Taler
KEYWORD: Privacy
SCIENTIFIC DESCRIPTION: Taler is a Chaum-style digital payment system that enables anonymous payments while ensuring that entities that receive payments are auditable. In Taler, customers can never defraud anyone, merchants can only fail to deliver the merchandise to the customer, and payment service providers can be fully audited. All parties receive cryptographic evidence for all transactions, still, each party only receives the minimum information required to execute transactions. Enforcement of honest behavior is timely, and is at least as strict as with legacy credit card payment systems that do not provide for privacy.

The key technical contribution underpinning Taler is a new refresh protocol which allows fractional payments and refunds while maintaining untraceability of the customer and unlinkability of transactions. The refresh protocol combines an efficient cut-and-choose mechanism with a link step to ensure that refreshing is not abused for transactional payments.

We argue that Taler provides a secure digital currency for modern liberal societies as it is a flexible, libre and efficient protocol and adequately balances the state's need for monetary control with the citizen's needs for private economic activity.
FUNCTIONAL DESCRIPTION: Taler is a new electronic payment system. It includes an electronic wallet for customers, a payment backend for merchants and the main payment service provider logic called the exchange. Taler offers Chaum-style anonymous payments for citizens, and income-transparency for taxability.

- Participants: Florian Dold, Gabor Toth, Hans Grothoff, Jeffrey Paul Burdges and Marcello Stanisci
- Partner: The GNU Project
- Contact: Hans Grothoff
- URL: http://taler.net/

6.5. HyLeak

Hybrid Analysis Tool for Information Leakage
KEYWORD: Information leakage

FUNCTIONAL DESCRIPTION: HyLeak is an evolution of the QUAIL tool, also developed by the TAMIS team. HyLeak divides the input program into (terminal) components and decides for each of them whether to analyze it using precise or statistical analysis, by applying heuristics that evaluate the analysis cost of each component. Then, HyLeak composes the analysis results of all components into an approximate joint probability distribution of the secret and observable variables in the program. Finally, the tool estimates the Shannon leakage and its confidence interval.

- Partner: AIST Tsukuba
- Contact: Fabrizio Biondi

6.6. SimFI

Tool for Simulation Fault injection
KEYWORDS: Fault injection - Fault-tolerance
FUNCTIONAL DESCRIPTION: Fault injections are used to test the robust and security of systems. We have developed SimFI, a tool that can be used to simulate fault injection attacks against binary files. SimFI is lightweight utility designed to be integrated into larger environments as part of robustness testing and fault injection vulnerability detection.

- Contact: Nisrine Jafri
- URL: https://github.com/nisrine/Fault-Injection-Tool

6.7. DaD

Data-aware Defense
KEYWORD: Ransomware
FUNCTIONAL DESCRIPTION: DaD is a ransomware countermeasure based on a file system minifilter driver. It is a proof of concept and in its present condition cannot be used as a replacement of the existing antivirus solutions. DaD detects randomness of the data by monitoring the write operations on the file system. We monitor all the userland threads, and also the whole file system (i.e., not restricted to Documents). It blocks the threads that exceed a specific threshold. The malicious thread is not killed, we only block its next I/O operations.

- Contact: Aurélien Palisse

6.8. MASSE

Modular Automated Syntactic Signature Extraction
KEYWORDS: Malware - Syntactic analysis
FUNCTIONAL DESCRIPTION: The Modular Automated Syntactic Signature Extraction (MASSE) architecture is a new integrated open source client-server architecture for syntactic malware detection and analysis based on the YARA, developed with Teclib'. MASSE includes highly effective automated syntactic malware detection rule generation for the clients based on a server-side modular malware detection system. Multiple techniques are used to make MASSE effective at detecting malware while keeping it from disrupting users and hindering reverse-engineering of its malware analysis by malware creators. MASSE integrates YARA in a distributed system able to detect malware on endpoint systems using YARA, analyze malware with multiple analysis techniques, automatically generate syntactic malware detection rules, and deploy the new rules to the endpoints. The MASSE architecture is freely available to companies and institutions as a complete, modular, self-maintained antivirus solution. Using MASSE, a security department can immediately update the rule database of the whole company, stopping an infection on its tracks and preventing future ones.

- Contact: Axel Legay

6.9. Behavioral Malware Analysis

KEYWORDS: Artificial intelligence - Malware - Automatic Learning - Concolic Execution

FUNCTIONAL DESCRIPTION: Our approach is based on artificial intelligence. We extract graphs from programs, that represent their behaviors. Such graphs are called system call dependency graphs (SCDGs). Our software learns to distinguish malware from cleanware on a large set of malwares and cleanwares. Whenever we want to analyze a new program, we extract its graphs and use the result of the training to decide whether the new program to analyze is a malware.

- Partner: Cisco
- Contact: Axel Legay
- URL: https://team.inria.fr/tamis/

6.10. VITRAIL - Visualisation Tool

Real-Time, Advanced, Immersive Visualization of Software / Visualizer
KEYWORD: Visualization of software
SCIENTIFIC DESCRIPTION: It is difficult for developers to explore and understand the source code of large programs, for example in objet-oriented languages programs featuring thousands of classes. Visualization methods based on daily life metaphors have thus been proposed. The VITRAIL Visualization tool (or VITRAIL Vizualizer) makes it possible to display, visualize and explore Java programs in a metaphorical way, using the city metaphor. An execution trace of the Java (byte)code provided by VITRAIL JBInstrace tool, is provided as input to VITRAIL Visualizer which displays a city-like metaphorical world showing the static structure of the code as well as some dynamic elements (calls).
FUNCTIONAL DESCRIPTION: This program makes it possible to displays, visualizes and explores Java programs in a metaphorical way (using the city metaphore). Useful for complex application developers/architects.

RELEASE FUNCTIONAL DESCRIPTION: Early release

- Participants: Damien Bodenes, Olivier Demengeon and Olivier Zendra
- Contact: Olivier Zendra
- URL: http://vitrail.loria.fr

6.11. VITRAIL 6 JBInsTrace

Real-Time, Advanced, Immersive Visualization of Software / Java Bytecode Instrumenter and Tracer
KEYWORDS: Execution trace - Profiling - Instrumentation - Bytecode - Java - Basic block
SCIENTIFIC DESCRIPTION: VITRAIL JBInsTrace is a program to instrument Java bytecode to trace its execution. The trace contains both static and dynamic information (calls). It is produced by intercepting the JVM class loader and replacing it by ours. Thus Java bytecode file are not modified, since instrumentation is performed on the fly, in memory. This makes it possible to instrument the whole program code, including libraries. Java source code is not needed. The trace which is then fed into our program VITRAIL Visualizer is an XML-like file.
FUNCTIONAL DESCRIPTION: VITRAIL JBInsTrace is a program to instrument Java bytecode files to trace their execution. The trace is then fed into our VITRAIL Visualizer tool.

- Participants: Olivier Zendra and Pierre Caserta
- Contact: Olivier Zendra
- URL: http://vitrail.loria.fr

6.12. Platforms

6.12.1. Malware'o'Matic

This LHS platform is dedicated to the collect, the categorization and the analyze of malware. We are currently interested in a specific kind of malware the ransomware. The platform grabs periodically samples of public data bases, executes the ransomware without virtualization on a victim PC and evaluate the implemented detection mechanisms. Once a ransomware has been executed the image of the OS is automatically restored and a new sample is evaluated. The platform is fully automatic and target Windows platforms (seven, W10) in both 32 bits and 64 bits versions. More recent developments can be seen in the LHS Activity Report.

6.12.2. Faustine

This LHS platform is dedicated to the EM fault injection experiments. It is composed of a motion table (XY), a pulse generator, an amplifier and a control PC. It injects EM pulses in a controlled way on a targeted device using an EM probe. It controls with a high precision the timing and the edges of the pulse. A recent development consists in adding a FPGA board to control the trigger in a more convenient and precise way. Then, the pulse can be triggered while a specific information is sent to the board under attack. More recent developments can be seen in the LHS Activity Report.