RESEARCH CENTER

FIELD
Algorithmics, Programming, Software and Architecture

Activity Report 2015

Section Software

Edition: 2016-03-21

ALGORITHMICS, COMPUTER ALGEBRA AND CRYPTOLOGY

1. ARIC Project-Team 5
2. CARAMEL Project-Team 7
3. CASCADE Project-Team (section vide) 10
4. CRYPT Team (section vide) 11
5. GALAAD2 Team 12
6. GEOMETRICA Project-Team 14
7. GRACE Project-Team 16
8. LFANT Project-Team 17
9. POLSYS Project-Team 20
10. SECRET Project-Team (section vide) 21
11. SPECFUN Project-Team 22
12. VEGAS Project-Team 24

ARCHITECTURE, LANGUAGES AND COMPILATION

13. ALF Project-Team 26
14. ATEAMS Project-Team 30
15. CAIRN Project-Team 32
16. CAMUS Team 36
17. COMPSYS Project-Team 40
18. CORSE Team 43
19. DREAMPAL Project-Team 45
20. POSTALE Team 46
21. TASC Project-Team 48

EMBEDDED AND REAL-TIME SYSTEMS

22. AOSTE Project-Team 50
23. CONVECS Project-Team 53
24. HYCOMES Team 56
25. MUTANT Project-Team 57
26. PARKAS Project-Team 61
27. POSET Team 66
28. SPADES Project-Team 67
29. TEA Project-Team 69

PROOFS AND VERIFICATION

30. ANTIQUE Project-Team 75
31. CELTIQUE Project-Team 80
32. DEDUCTEAM Team 82
33. ESTASYS Team 86
34. GALLIUM Project-Team 88
35. MARELLE Project-Team 90
36. MEXICO Project-Team 92
37. PARSIFAL Project-Team 93

38. PI.R2 Project-Team 95
39. SUMO Project-Team 97
40. TOCCATA Project-Team 99
41. VERIDIS Project-Team 102

SECURITY AND CONFIDENTIALITY

42. CARTE Project-Team 105
43. CASSIS Project-Team 106
44. COMETE Project-Team 109
45. DECENTRALISE Team 110
46. DICE Team 111
47. PRIVATICS Project-Team 112
48. PROSECCO Project-Team 114

ARIC Project-Team

6. New Software and Platforms

6.1. FPLLL: a lattice reduction library

fplll contains several algorithms on lattices that rely on floating-point computations. This includes implementations of the floating-point LLL reduction algorithm, 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 also includes a rigorous floating-point implementation of the Kannan-Fincke-Pohst algorithm that finds a shortest non-zero lattice vector, and the BKZ reduction algorithm.

The fplll library is distributed under the LGPL license. It has been used in or ported to several mathematical computation systems such as Magma, Sage, and PariGP. It is also used for cryptanalytic purposes, to test the resistance of cryptographic primitives.

- Participants: Shi Bai, Damien Stehlé
- Contact: Damien Stehlé
- URL: https://github.com/dstehle/fplll

6.2. GNU MPFR: a library for arbitrary precision floating-point arithmetic

KEYWORDS: Multiple-Precision - Floating-point - Correct Rounding

GNU MPFR is an efficient multiple-precision floating-point library written in C 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. It is distributed under the LGPL license.

The development of MPFR started in Loria (Nancy). When Vincent Lefèvre moved from Nancy to Lyon, it became a joint project between the project-team Caramel (Nancy) and AriC. Many systems use MPFR, several of them being listed on its web page. MPFR 3.1.3 was released on 19 June 2015.

New developments in the trunk: Full rewrite of `mpfr_sum` completed, with new tests [38]. Generic tests improved. Bug fixes and various improvements, in particular concerning the flags.

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

6.3. Gfun: a Maple package for solutions of linear differential or recurrence equations

Gfun is a Maple package that provides tools for: guessing a sequence or a series from its first terms; manipulating rigorously solutions of linear differential or recurrence equations, using the equation as a data-structure.

Its development moved to AriC with Bruno Salvy in 2012, while a submodule NumGfun dedicated to symbolic-numeric computations with linear ODEs has been developed by Marc Mezzarobba during his post-doc at AriC. An old version of gfun is distributed with the Maple library. Newer versions are available on the web page of gfun, which also lists a number of articles by scientists who cited it.

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

6.4. Sipe: a library for very low precision computations with correct rounding

KEYWORDS: Floating-point - Correct Rounding

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. It is distributed under the LGPL license and mostly used internally.

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

6.5. LinBox: a C++ library for exact, high-performance linear algebra computation

LinBox is a C++ template library for exact, high-performance linear algebra computation with dense, sparse, and structured matrices over the integers and over finite fields. LinBox is distributed under the LGPL license. The library is developed by a consortium of researchers in Canada, USA, and France. Clément Pernet is a main contributor, especially with a focus on parallel aspects during the period covered by this report.

- Participant:
- Contact: Clément Pernet
- URL: http://www.linalg.org

6.6. Exhaustive Tests for the Correct Rounding of Mathematical Functions

Participant: Vincent Lefèvre.

The search for the worst cases for the correct rounding (hardest-to-round cases) of mathematical functions ($\exp$, $\log$, $\sin$, $\cos$, etc.) in a fixed precision (mainly double precision) using Lefèvre's algorithm is implemented by a set of utilities written in Perl, with calls to Maple/intpakX for computations on intervals and with C code generation for fast computations. It also includes a client-server system for the distribution of intervals to be tested and for tracking the status of intervals (fully tested, being tested, aborted).

The support for the $\tanh$ function has been added, and this function has been tested on the full domain (together with its inverse function). Results are available from: https://www.vinc17.net/research/testlibm/

- Participant: Vincent Lefèvre
- Contact: Vincent Lefèvre

6.7. Multiplication by Integer Constants

Participant: Vincent Lefèvre.

A Perl implementation of algorithms for the multiplication by integer constants has been updated to get more results based on exhaustive tests: threading has been implemented in this part of the script.

- Participant: Vincent Lefèvre
- Contact: Vincent Lefèvre
- URL: https://www.vinc17.net/research/mulbyconst/#patterns

CARAMEL Project-Team

6. New Software and Platforms

6.1. Belenios

Belenios - Verifiable online voting system
KEYWORD: E-voting
FUNCTIONAL DESCRIPTION

In collaboration with the CASSIS team, we develop an open-source private and verifiable electronic voting protocol, named BELENIOS. Our system is an evolution of an existing system, Helios, developed by Ben Adida, and used e.g., by UCL and the IACR association in real elections. The main differences with Helios are the following ones:

- In Helios, the ballot box publishes the encrypted ballots together with their corresponding voters. This raises a privacy issue in the sense that whether someone voted or not shall not necessarily be publicized on the web. Publishing this information is in particular forbidden by CNIL's recommendation. BELENIOS no longer publishes voters' identities, still guaranteeing correctness of the tally.
- Helios is verifiable except that one has to trust that the ballot box will not add ballots. The addition of ballots is particularly hard to detect as soon as the list of voters is not public. We have therefore introduced an additional authority that provides credentials that the ballot box can verify but not forge [27].

This new version has been implemented by Stéphane Glondu [0]. The first public release has been done in January 2014. Belenios has been used in Sep 2015 for the election of the new leader of the GT-C2 (Groupe de Travail Codes et Cryptographie) which is part of the GdR-IM (Groupement de Recherche Informatique Mathématique). The GT calcul formel of the GdR-IM plans to use Belenios in 2016 for the election of its new leader.

An online platform [0] has been released in September 2015, so that setting up a new election can be done entirely from within a browser.

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

6.2. CADO-NFS

Crible Algébrique: Distribution, Optimisation - Number Field Sieve
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: Emmanuel Thomé, Pierrick Gaudry, Paul Zimmermann, Alexander Kruppa, François Morain, Cyril Bouvier.
- Contact: Emmanuel Thomé
- URL: http://cado-nfs.gforge.inria.fr/

[0] http://belenios.gforge.inria.fr/
[0] https://belenios.loria.fr/

In December 2015, a major new release of CADO-NFS, version 2.2.0, was published. It contains several bug fixes, efficiency improvements, and the computation of discrete logarithms is now almost "push-button".

6.3. CMH

Computation of Igusa Class Polynomials
KEYWORDS: Mathematics - Cryptography - Number theory
FUNCTIONAL DESCRIPTION

Cmh computes Igusa class polynomials, parameterizing two-dimensional abelian varieties (or, equivalently, Jacobians of hyperelliptic curves of genus 2) with given complex multiplication.

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

6.4. GF2X

FUNCTIONAL DESCRIPTION

GF2X is a software library for polynomial multiplication over the binary field, developed together with Richard Brent (Australian National University, Canberra, Australia). It holds state-of-the-art implementation of fast algorithms for this task, employing different algorithms in order to achieve efficiency from small to large operand sizes (Karatsuba and Toom-Cook variants, and eventually Schönhage's or Cantor's FFT-like algorithms). FG2X takes advantage of specific processor instructions (SSE, PCLMULQDQ).

- Participants: Pierrick Gaudry, Emmanuel Thomé and Paul Zimmermann
- Contact: Emmanuel Thomé
- URL: https://gforge.inria.fr/projects/gf2x/

6.5. GNU MPC

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.

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

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

6.7. MPFQ

FUNCTIONAL DESCRIPTION

MPFQ is (yet another) library for computing in finite fields, with automatic generation of code for fields known at compile-time. It consists of roughly 18,000 lines of Perl code, which generate most of the C code. MPFQ is used in CADO-NFS, in particular for the linear algebra step during discrete logarithm computations.

- Participants: Emmanuel Thomé, Pierrick Gaudry and Luc Sanselme
- Contact: Pierrick Gaudry
- URL: http://mpfq.gforge.inria.fr/

6.8. Tinygb

Tinygb is a small software tool written in C++. Its aim is to provide an interface between several existing libraries (finite field arithmetic, linear algebra) for Gröbner bases computations occurring in problems investigated by the CARAMEL group. The focus is not on the efficiency of the implementation, since this is already successfully achieved in other existing software such as *FGb* (developed by Jean-Charles Faugère) or in the CAS Magma (Gröbner bases algorithms are implemented by Alan Steel). The goal of Tinygb is to be a flexible research tool where variants of classical algorithms can be tested. Tinygb is still in development since it requires more testing and packaging before being released.

- Participants: Pierre-Jean Spaenlehauer

6.9. Platforms

6.9.1. CATREL cluster

Installed in 2013, the CATREL computer cluster now plays an essential role in providing the team with the necessary resources to achieve significant computations, which illustrate well the efficiency of the algorithms developed in our research, together with their implementations.

In 2015, the CATREL cluster was in particular used for the precomputations performed for the LOGJAM attack [15]. It was the main computing resource for a record discrete logarithm computation in finite fields of the form $\mathbb{F}_{p^3}$ of 512 bits, and a larger computation for this kind of fields is currently running. It was also used intensively to optimize the sieving parameters of CADO-NFS for factoring numbers from 60 to 155 digits, in the preparation of the release 2.2.0. It was used to factor 47 large integers from nine aliquot sequences starting from 276 to 204828, the largest one being a 190-digit composite number from sequence 660. The current largest element known from an aliquot sequence has 197 digits (sequence 19560), and we expect the 200-digit frontier will be reached in 2016. Several experiments were also made with variations of the polynomial selection algorithm from [10] on RSA-896 and RSA-1024.

CASCADE Project-Team (section vide)

CRYPT Team (section vide)

GALAAD2 Team

5. New Software and Platforms

5.1. AXEL

KEYWORDS: CAO - Algebraic geometric modeler
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).

5.2. Mathemagix

SCIENTIFIC DESCRIPTION

The project aims at building a bridge between symbolic computation and numerical analysis. It is structured by collaborative software developments of different groups in the domain of algebraic and symbolic-numeric computation.

In this framework, we are working more specifically on the following components:

realroot: a set of solvers using subdivision methods to isolate the roots of polynomial equations in one or several variables, continued fraction expansion of roots of univariate polynomials, Bernstein basis representation of univariate and multivariate polynomials and related algorithms, exact computation with real algebraic numbers, sign evaluation, comparison, certified numerical approximation.

shape: tools to manipulate curves and surfaces of different types including parameterized, implicit with different type of coefficients, algorithms to compute their topology, intersection points or curves, self-intersection locus, singularities, ...

These packages are integrated from the former library Synaps (SYmbolic Numeric APplicationS) dedicated to symbolic and numerical computations. There are also used in the algebraic-geometric modeler axel .
FUNCTIONAL DESCRIPTION

Mathemagix is a free computer algebra system which consists of a general purpose interpreter, which can be used for non-mathematical tasks as well, and efficient modules on algebraic objects. It includes the development of standard libraries for basic arithmetic on dense and sparse objects (numbers, univariate and multivariate polynomials, power series, matrices, etc., based on FFT and other fast algorithms). These developments, based on C++, offer generic programming without losing effectiveness, via the parameterization of the code (template) and the control of their instantiations.

- Participants: Bernard Mourrain, Grégoire Lecerf, Philippe Trebuchet and Joris Van Der Hoeven
- Contact: Bernard Mourrain
- URL: http://www.mathemagix.org/

GEOMETRICA Project-Team

6. New Software and Platforms

6.1. GUDHI

Geometric Understanding in Higher Dimensions

Scientific 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.

Functional Description

The current release of the GUDHI library includes: – Data structures to represent, construct and manipulate simplicial complexes. – Algorithms to compute persistent homology and multi-field persistent homology. – Simplification methods via implicit representations. - A graphical user interface and several examples and datasets.
It also has improved performance, portability and documentation.

- Participants: Jean-Daniel Boissonnat, Marc Glisse, Anatole Moreau, Vincent Rouvreau and David Salinas
- Contact: Jean-Daniel Boissonnat
- URL: https://project.inria.fr/gudhi/software/

6.2. CGAL dD Triangulations

CGAL module: Triangulations in any dimension

Keywords: Triangulation - Delaunay triangulation

Functional Description

This package of CGAL (Computational Geometry Algorithms Library, http://www.cgal.org) allows to compute triangulations and Delaunay triangulations in any dimension. Those triangulations are built incrementally and can be modified by insertion or removal of vertices.

- Participants: Samuel Hornus, Olivier Devillers and Clément Jamin
- Contact: Clément Jamin
- URL: http://doc.cgal.org/4.6/Triangulation/

6.3. CGAL Kernel_d

CGAL module: High-dimensional kernel Epick_d

Functional Description

Several functions were added in release 4.7 in preparation for a future alpha-complex implementation.

- Participants: Marc Glisse
- Contact: Marc Glisse
- URL: http://doc.cgal.org/4.7/Kernel_d/

6.4. R package TDA

Topological Data Analysis package for the R software

FUNCTIONAL DESCRIPTION

the R package TDA provides some tools for Topological Data Analysis. In particular, it includes implementations of functions that, given some data, provide topological information about the underlying space, such as the distance function, the distance to a measure, the kNN density estimator, the kernel density estimator, and the kernel distance.

- Participants: Clément Maria, Vincent Rouvreau
- Contact: Vincent Rouvreau
- URL: https://cran.r-project.org/web/packages/TDA/index.html

6.5. cgal Periodic Triangulations and Meshes

The CGAL library offers a package to compute the 3D periodic Delaunay triangulation of a point set in $\mathbb{R}^3$, more precisely the Delaunay triangulation of a point set in the 3-dimensional flat torus with cubic domain [49]. The package has been used in various fields. [0]

We have been extending this package in three directions:

First, a few new small functions have been added to the Delaunay triangulation class and integrated in CGAL 4.7.

We have developed and documented some new classes allowing to compute *weighted* periodic Delaunay triangulations. They have been submitted to the CGAL editorial board and accepted for inclusion in CGAL. The code still needs some polishing, and the testsuite must be completed, before a public distribution in CGAL.

We have continued our work to use this package together with the 3D mesh generation package of CGAL [48], in order to propose a construction of meshes of periodic volumes. Although last year's preliminary results were already convincing [50], [51], the work is not ready yet for being submitted to CGAL: the code requires to be completed, documented, and extensively tested.

- Participant : Aymeric Pellé
- Contact: Monique Teillaud (Vegas project-team)
- This work was done in the framework of the Inria ADT *(Action de Développement Technologique)* OrbiCGAL http://www.loria.fr/~teillaud/ADT-OrbiCGAL/

[0] see http://www.cgal.org/projects.html

GRACE Project-Team

6. New Software and Platforms

6.1. 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/

6.2. Platforms

6.2.1. 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 had two directions for improvement:

1. renewing the APIs to make them actually usable by researchers, and
2. incorporating efficient programs for decoding, like J. Nielsen's CodingLib, which contains many new algorithms.

After a year on the project, which started October 1st, 2014, we have been able to completely rethink and rewrite the API to a new structure able to support many mathematical constructions and integrate it in Sage. We also implemented numerous code classes and decoding algorithms, including cyclic codes over any finite field and list decoding of GRS codes, which are not available in Maple, Magma and Mathematica. As integrating code in Sage is a slow process, which requires external developers, we attended two Sage workshops (Sage Days 66 in Liège and Sage Days 70 in Berkeley) and welcomed one at Inria Saclay http://wiki.sagemath.org/GroupeUtilisateursParis#mercredi-1er-juillet-2015-module-de-codage-actis-pour-sage to spread the word on the project and meet the main Sage developers. We're now trusted members of the community, and we were able to integrate several patches in Sage.

- Contact: David Lucas
- URL: https://bitbucket.org/lucasdavid/sage_coding_project/wiki/Home
- One can check a full list of accepted and pending ACTIS patches for Sage here : http://trac.sagemath.org/ticket/18846.

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. Arb

FUNCTIONAL DESCRIPTION

Arb is a C library for arbitrary-precision floating-point ball arithmetic. It supports real and complex numbers, polynomials, power series, matrices, and evaluation of many transcendental functions. All is done with automatic, rigorous error bounds. It has been accepted for inclusion in SageMath.

- Participant: Fredrik Johansson
- Contact: Fredrik Johansson
- URL: http://fredrikj.net/arb/

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

5.4. CM

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.

- Participant: Andreas Enge
- Contact: Andreas Enge
- URL: http://www.multiprecision.org/index.php?prog=cm&page=home

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

5.6. 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-bordeaux.fr/~belabas/research/software/cubic-1.2.tgz

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

5.8. GNU MPC

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.

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

5.9. 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

5.10. MPFRCX

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.

- Participant: Andreas Enge
- Contact: Andreas Enge
- URL: http://www.multiprecision.org/index.php?prog=mpfrcx

5.11. Nemo

FUNCTIONAL DESCRIPTION Nemo is a computer algebra package for the Julia programming language maintained by William Hart with code by William Hart, Tommy Hofmann, Claus Fieker, Fredrik Johansson, Oleksandr Motsak).

The features of Nemo include multiprecision integers and rationals, integers modulo n, p-adic numbers, finite fields (prime and non-prime order), number field arithmetic, maximal orders of number fields, arithmetic of ideals in maximal orders, arbitrary precision real and complex balls, generic polynomials, power series, fraction fields, residue rings and matrices.

- Participant: Fredrik Johansson
- Contact: William Hart
- URL: http://nemocas.org/

5.12. PARI/GP

FUNCTIONAL DESCRIPTION

PARI/GP is a widely used computer algebra system designed for fast computations in number theory (factorisation, algebraic number theory, elliptic curves, ...), 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: Karim Belabas, Henri Cohen, Andreas Enge and Hamish Ivey-Law
- Contact: Karim Belabas
- URL: http://pari.math.u-bordeaux.fr/

POLSYS Project-Team

5. New Software and Platforms

5.1. FGb

FUNCTIONAL DESCRIPTION

FGb is a powerful software for computing Groebner 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 Faugère
- Contact: Jean-Charles Faugère
- URL: http://polsys.lip6.fr/~jcf/Software/FGb/index.html

5.2. FGb Light

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

5.3. 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: Brice Boyer
- URL: http://www-polsys.lip6.fr/~jcf/Software/index.html

5.4. RAGlib

FUNCTIONAL DESCRIPTION

RAGLib is a Maple library for solving over the reals polynomial systems and computing sample points in semi-algebraic sets.

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

5.5. 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 $\sim 5\,000$ lines.

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

SECRET Project-Team (section vide)

SPECFUN Project-Team

5. New Software and Platforms

5.1. Coq

KEYWORDS: Proof - Certification - Formalisation
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, ...

- Participants: Benjamin Grégoire, Enrico Tassi, Bruno Barras, Yves Bertot, Pierre Boutillier, Xavier Clerc, Pierre Courtieu, Maxime Denes, Stéphane Glondu, Vincent Gross, Hugo Herbelin, Pierre Letouzey, Assia Mahboubi, Julien Narboux, Jean-Marc Notin, Christine Paulin-Mohring, Pierre-Marie Pédrot, Loïc Pottier, Matthias Puech, Yann Régis-Gianas, François Ripault, Matthieu Sozeau, Arnaud Spiwack, Pierre-Yves Strub, Benjamin Werner, Guillaume Melquiond and Jean-Christophe Filliâtre
- Partners: CNRS - Université Paris-Sud - ENS Lyon - Université Paris-Diderot
- Contact: Hugo Herbelin
- URL: http://coq.inria.fr/

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

5.3. 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: Stéphanie Petit, Alexis Darrasse, Frédéric Chyzak and Maxence Guesdon
- Contact: Frédéric Chyzak
- URL: http://algo.inria.fr/encyclopedia/

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.

- Participants: Andrea Asperti, Jeremy Avigad, Yves Bertot, Cyril Cohen, François Garillot, Georges Gonthier, Stéphane Le Roux, Assia Mahboubi, Sidi Ould Biha, Ioana Pasca, Laurence Rideau, Alexey Solovyev, Enrico Tassi and Russell O'connor
- Contact: Assia Mahboubi
- URL: http://www.msr-inria.fr/projects/mathematical-components-2/

5.5. Ring

Functional Description

Coq normalization tool and decision procedure for expressions in commutative ring theories. Implemented in Coq and OCaml. Integrated in the standard distribution of the Coq proof assistant since 2005.

- Contact: Assia Mahboubi

5.6. Ssreflect

Functional Description

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

- Participants: Cyril Cohen, Yves Bertot, Laurence Rideau, Enrico Tassi, Laurent Théry, Assia Mahboubi and Georges Gonthier
- Contact: Yves Bertot
- URL: http://ssr.msr-inria.inria.fr/

VEGAS Project-Team

5. New Software and Platforms

5.1. ISOTOP

Topology and geometry of planar algebraic curves
KEYWORDS: Topology - Curve plotting - Geometric computing

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, registered at the APP in June 2011, has been developed since 2007 in collaboration with F. Rouillier from Inria Paris - Rocquencourt. The distributed version is based on the method described in [3], which presents several improvements over previous methods. In particular, our approach does not require generic position. This version is competitive with other implementations (such as ALCIX and INSULATE developed at MPII Saarbrücken, Germany and TOP developed at Santander Univ., Spain). It performs similarly for small-degree curves and performs significantly better for higher degrees, in particular when the curves are not in generic position.

We are currently working on an improved version integrating a new bivariate polynomial solver based on several of our recent results published in [11], [22], [27]. This version is not yet distributed.

- Contact: Sylvain Lazard & Marc Pouget
- URL: http://vegas.loria.fr/isotop/

5.2. SubdivisionSolver

KEYWORDS: Numerical solver - Polynomial or analytical systems

The software SubdivisionSolver solves square systems of analytic equations on a compact subset of a real space of any finite dimension. SubdivisionSolver is a numerical solver and as such it requires that the solutions in the subset are isolated and regular for the input system (i.e. the Jacobian must not vanish). SubdivisionSolver is a subdivision solver using interval arithmetic and multiprecision arithmetic to achieve certified results. If the arithmetic precision required to isolate solutions is known, it can be given as an input parameter of the process, otherwise the precision is increased on-the-fly. In particular, SubdivisionSolver can be interfaced with the Fast_Polynomial library (https://bil.inria.fr/en/software/view/2423/tab) to solve polynomial systems that are large in terms of degree, number of monomials and bit-size of coefficients.

The software is based on a classic branch and bound algorithm using interval arithmetic: an initial box is subdivided until its sub-boxes are certified to contain either no solution or a unique solution of the input system. Evaluation is performed with a centered evaluation at order two, and existence and uniqueness of solutions is verified thanks to the Krawczyk operator.

SubdivisionSolver uses two implementations of interval arithmetic: the C++ boost library that provides a fast arithmetic when double precision is enough, and otherwise the C mpfi library that allows to work in arbitrary precision. Considering the subdivision process as a breadth first search in a tree, the boost interval arithmetic is used as deeply as possible before a new subdivision process using higher precision arithmetic is performed on the remaining forest.

We used SubdivisionSolver for the experiments in [26], [14], see Section 6.3.2 .

- Contact: Rémi Imbach
- URL: https://bil.inria.fr/fr/software/view/2605/tab

5.3. CGAL Periodic Triangulations and Meshes

The CGAL library offers a package to compute the 3D periodic Delaunay triangulation of a point set in $\mathbb{R}^3$, more precisely the Delaunay triangulation of a point set in the 3-dimensional flat torus with cubic domain [30]. The package has been used in various fields. [0]

We have been extending this package in three directions:

First, a few new small functions have been added to the Delaunay triangulation class and integrated in CGAL 4.7.

We have developed and documented some new classes allowing to compute *weighted* periodic Delaunay triangulations. They have been submitted to the CGAL editorial board and accepted for inclusion in CGAL. The code still needs some polishing, and the testsuite must be completed, before a public distribution in CGAL.

We have continued our work to use this package together with the 3D mesh generation package of CGAL [29], in order to propose a construction of meshes of periodic volumes. Although last year's preliminary results were already convincing [32], [33], the work is not ready yet for being submitted to CGAL: the code requires to be completed, documented, and extensively tested.

- Contact: Monique Teillaud
- In collaboration with Aymeric Pellé (Geometrica project-team)
- This work was done in the framework of the Inria ADT *(Action de Développement Technologique)* OrbiCGAL http://www.loria.fr/~teillaud/ADT-OrbiCGAL/

[0] see http://www.cgal.org/projects.html

ALF Project-Team

6. New Software and Platforms

6.1. ATC

Address Trace Compression
KEYWORDS: Compressing - Decompressing - Address traces
FUNCTIONAL DESCRIPTION

ATC is a utility and a C library for compressing/decompressing address traces. It implements a new lossless transformation, Bytesort, that exploits spatial locality in address traces. ATC leverages existing general-purpose compressors such as gzip and bzip2. ATC also provides a lossy compression mode that yields higher compression ratios while preserving certain important characteristics of the original trace.

- Participant: Pierre Michaud
- Contact: Pierre Michaud
- URL: https://team.inria.fr/alf/software/atc/

6.2. ATMI

Modeling microprocessor 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.

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

6.3. Barra

Modelisation of a GPU architecture
KEYWORDS: Simulator - GPU - Computer architecture
SCIENTIFIC DESCRIPTION

Research on throughput-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 simulates CUDA programs at the assembly language level (Tesla ISA). Its ultimate goal is to provide a 100 % bit-accurate simulation, offering bug-for-bug compatibility with NVIDIA G80-based GPUs. It works directly with CUDA executables, neither source modification nor recompilation is required.

Barra is primarily intended as a tool for research in computer architecture, although it can also be used to debug, profile and optimize CUDA programs at the lowest level.

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

6.4. HEPTANE

Static analyser of Worst-Case Execution Time
KEYWORD: WCET
FUNCTIONAL DESCRIPTION

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.
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.

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

6.5. If-memo

KEYWORD: Performance, function memoization, dynamic optimization
Status: Ongoing development, early prototype. Registered with APP (Agence de Protection des Programmes) under number IDDN.FR.001.250013.000.S.P.2015.000.10800.

SCIENTIFIC DESCRIPTION

Memoization is the technique of saving result of executions so that future executions can be omitted when the inputs repeat. Memoization has been proposed in previous literature at the instruction level, basic block level and function level using hardware as well as pure software level approaches including changes to programming language.

We proposed software memoization of pure functions for procedural languages. We rely on the operating system loader, taking advantage of the LD_PRELOAD feature of UNIX systems. By setting this variable to the path of a shared library, we instruct the loader to first look to missing symbols in that library. Our library redefines the functions we wish to intercept. The interception code is very straightforward: it receives the same parameter as the target function and checks in a table (a software cache) if this value is readily available. In the favorable case, the result value is immediately returned. Otherwise, we invoke the original function, and store the result in the cache before returning it.

Our technique does not require the availability of source code and thus can be applied even 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. We validated If-memo with x86-64 platform using both GCC and icc compiler tool-chains, and ARM cortex-A9 platform using GCC.

- Participants: Erven Rohou and Arjun Suresh
- Contact: Erven Rohou

6.6. Padrone

KEYWORDS: Legacy code - Optimization - Performance analysis - Dynamic Optimization
Status: Registered with APP (Agence de Protection des Programmes) under number IDDN.FR.001.250013.000.S.P.2015.000.1080

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

6.7. STiMuL

Steady temperature in Multi-Layers components
FUNCTIONAL DESCRIPTION

STiMuL is a C library for modeling steady-state heat conduction in microprocessors. It can be used to obtain temperature from power density or power density from temperature. It can also be used to model stacked dies. STiMuL does not model time-varying temperature. For time-varying temperature, other models must be used, such as ATMI.

- Participant: Pierre Michaud
- Contact: Pierre Michaud
- URL: https://team.inria.fr/alf/software/stimul/

6.8. TPCalc

Throughput calculator
KEYWORDS: Architecture - Performance analysis
FUNCTIONAL DESCRIPTION

TPCalc is a throughput calculator for microarchitecture studies concerned with multi-program workloads consisting of sequential programs. Because microarchitecture simulators are slow, it is difficult to simulate throughput experiments where a multicore executes many jobs that enter and leave the system. The usual practice of measuring instantaneous throughput on independent coschedules chosen more or less randomly is not a rigorous practice because it assumes that all the coschedules are equally important, which is not always true. TPCalc can compute the average throughput of a throughput experiment without actually doing the throughput experiment. The user first defines the workload heterogeneity (number of different job types), the multicore configuration (number of cores and symmetries). TPCalc provides a list of base coschedules. The user then simulates these coschedules, using some benchmarks of his choice, and feeds back to TPCalc the measured execution rates (e.g., instructions per cycle or instructions per second).TPCalc eventually outputs the average throughput.

- Participant: Pierre Michaud
- Partner: Ghent University
- Contact: Pierre Michaud
- URL: http://www.irisa.fr/alf/downloads/michaud/tpcalc.html

6.9. tiptop

KEYWORDS: Performance, hardware counters, analysis tool.
SCIENTIFIC DESCRIPTION

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, released June 2015.

Tiptop is a new simple and flexible user-level tool that collects hardware counter data on Linux platforms (version 2.6.31+). Tiptop has been integrated in major Linux distributions, such as Fedora, Debian, Ubuntu. FUNCTIONAL DESCRIPTION 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* — non-privileged users can only watch processes they own, ability to monitor anybody's process opens the door to side-channel attacks.
- The usage is similar to *top*. There is no need for the source code of the applications of interest, making it possible to monitor proprietary applications or libraries. And since there is no probe to insert in the application, understanding of the structure and implementation of complex algorithms and code bases is not required.
- Applications do not need to be restarted, and monitoring can start at any time (obviously, only events that occur after the start of *tiptop* are observed).
- Events can be counted per thread, or per process.
- Any expression can be computed, using the basic arithmetic operators, constants, and counter values.
- A configuration file lets users define their prefered setup, as well as custom expressions.
- Participant: Erven Rohou
- Contact: Erven Rohou
- URL: http://tiptop.gforge.inria.fr

6.10. Parasuite

Participants: Sylvain Collange, Thibault Person, Erven Rohou, André Seznec.

Parasuite: parallel benchmarks for multi-core CPUs, clusters and accelerators

Despite the ubiquity of parallel architectures in all computing segments, the research community often lacks benchmarks representative of parallel applications. The Inria Parallel Benchmark Suite (Parasuite) seeks to address this need by providing a set of representative parallel benchmarks for the architecture, compiler and system research communities. Parasuite targets the main contemporary parallel programming technologies: shared-memory multi-thread parallelism for multi-core, message-passing parallelism for clusters and fine-grained data-level parallelism for GPU architectures and SIMD extensions.

All benchmarks come with input datasets of various sizes, to accommodate use cases ranging from microarchitecture simulation to large-scale performance evaluation. Correctness checks on the computed results enable automated regression testing. In order to support computer arithmetic optimization and approximate computing research scenarios, the correctness checks favor accuracy metrics evaluating domain-specific relevance rather than bit-exact comparisons against an arbitrary reference output.

Visit http://parasuite.inria.fr/

ATEAMS Project-Team

5. New Software and Platforms

5.1. MicroMachinations

FUNCTIONAL DESCRIPTION

Objective: To create an integrated, live environment for modelling and evolving game economies. This will allow game designers to experiment with different strategies to realise game mechanics. The environment integrates with the SPIN model checker to prove properties (reachability, liveness). A runtime system for executing game economies allows MicroMachinations models to be embedded in actual games.

Impact: One of the important problems in game software development is the distance between game design and implementation in software. MicroMachinations has the potential to bridge this gap by providing live design tools that directly modify or create the desired software behaviours.

- Participants: Paul Klint and Riemer Van Rozen
- Contact: Riemer Van Rozen
- URL: https://github.com/vrozen/MM-Lib

5.2. OSSMETER

KEYWORDS: Software Quality, Metrics, Open-source SCIENTIFIC DESCRIPTION: OSSMETER meets the challenge of software project quality assessment via fact-based business intelligence. The goal of the project was to design and evaluate a platform for incremental analysis of long lasting open-source projects to support decision making on the corporate level. FUNCTIONAL DESCRIPTION: OSSMETER is a platform which integrates metrics of open-source projects: their source code quality, the contents of their social interactions and their activity in issue tracking systems. It includes a fully programmable user-defined quality model utility and configurable dash-board user-interface. The basic metrics of the platform and their aggregation to the project level are carefully considered and rationalised.

- Participants: Paul Klint, Jurgen Vinju, Tijs Van Der Storm, Ashim Shahi, Bas Basten.
- Contact: Jurgen Vinju
- URL: http://www.ossmeter.org/

5.3. Rascal

KEYWORDS: Metaprogramming - Language
SCIENTIFIC DESCRIPTION

Rascal primitives include immutable data, context-free grammars and algebraic data-types, relations, relational calculus operators, advanced patterns matching, generic type-safe traversal, comprehensions, concrete syntax for objects, lexically scoped backtracking, and string templates for code generation. It has libraries for integrating language front-ends, for reusing analysis algorithms, for getting typed meta-data out of version management systems, for interactive visualization, etc.

FUNCTIONAL DESCRIPTION

Rascal is a programming language, such that meta programs can be created by, understood by, and debugged by programmers.

You want to use the best tool for the job when analyzing, transforming or generating source code, so normally you will end up with many different tools, possibly even written in different languages. Now the problem is to integrate these tools again. Rascal solves this problem by integrating source code analysis, transformation, and generation primitives on the language level. Use it for any kind of metaprogramming task: to construct parsers for programming languages, to analyze and transform source code, or to define new DSLs with full IDE support.

- Participants: Paul Klint, Jurgen Vinju, Tijs Van Der Storm, Davy Landman, Bert Lisser, Atze Van Der Ploeg, Vadim Zaytsev, Anastasia Izmaylova, Michael Steindorfer, Jouke Stoel, Ali Afroozeh and Ashim Shahi
- Contact: Paul Klint
- URL: http://www.rascal-mpl.org/

5.4. Meerkat

FUNCTIONAL DESCRIPTION

Objective: To enable fully context-free general parsing using a parser combinator library (including allowing left recursion and arbitrary context-sensitive disambiguation).

Impact: Meerkat explores algorithmic advances in context-free general parsing (based on the GLL parsing algorithm and memoized continuations) in the context of a scala parsing combinator library. This library uniquely combines the worst-case execution time guarantees of GLL with the flexibility of parsing combinators. [47]

- Participants: Anastasia Izmaylova, Ali Afroozeh and Tijs van der Storm.
- Contact: Anastasia Izmaylova, Ali Afroozeh
- URL: http://meerkat-parser.github.io/

5.5. Iguana

FUNCTIONAL DESCRIPTION

Objective: To provide a data-dependent context-free general parsing infra-structure for parsing programming languages and other formal data, program and modeling notations.

Impact: Iguana is a fast implementation of data-dependent grammars based on the GLL context-free parsing algorithm with data-dependent non-terminals and constraints on top. It comes with a number of high-level disambiguation constructs which are translated to the intermediate layer of data-dependent (E)BNF before being loaded into an object-oriented implementation of GLL based on abstract transition network. Using Iguana parsers for languages which are considered to be hard to parse (such as Haskell and OCAML) are within reach of being generated from simple declarative specifications [25].

- Participants: Anastasia Izmaylova, Ali Afroozeh.
- Contact: Anastasia Izmaylova, Ali Afroozeh
- URL: http://iguana-parser.github.io/

5.6. Capsule

FUNCTIONAL DESCRIPTION

Objective: A generic and highly optimised product-family of immutable collection data-structures.

Impact: Capsule is a library for immutable sets, maps and tables. The code is generated using high-level descriptions of the requirements and internal trade-offs of hash-trie map based implementations. We are using this code generator to experiment with the fastest and leanest representations of these persistent data-types to satisfy the requirements of Rascal meta-programming applications in static analysis, empirical research in software engineering and software analytics [37].

- Participants: Michael Steindorfer, Jurgen Vinju
- Contact: Michael Steindorfer
- URL: http://usethesource.io/projects/capsule/

CAIRN Project-Team

6. New Software and Platforms

6.1. Panorama

With the ever raising complexity of embedded applications and platforms, the need for efficient and customizable compilation flows is stronger than ever. This need of flexibility is even stronger when it comes to research compiler infrastructures that are necessary to gather quantitative evidence of the performance/energy or cost benefits obtained through the use of reconfigurable platforms. From a compiler point of view, the challenges exposed by these complex reconfigurable platforms are quite significant, since they require the compiler to extract and to expose an important amount of coarse and/or fine grain parallelism, to take complex resource constraints into consideration while providing efficient memory hierarchy and power management.

Because they are geared toward industrial use, production compiler infrastructures do not offer the level of flexibility and productivity that is required for compiler and CAD tool prototyping. To address this issue, we have designed an extensible source-to-source compiler infrastructure that takes advantage of leading edge model-driven object-oriented software engineering principles and technologies.

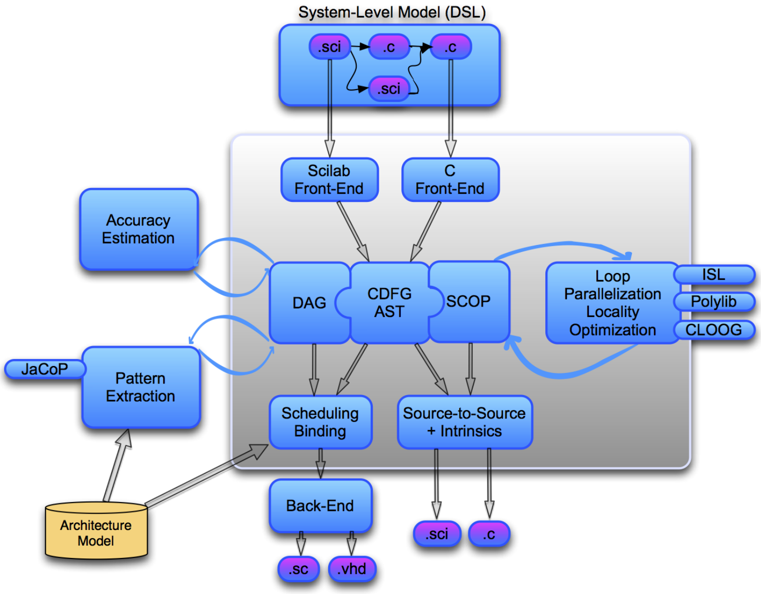


Figure 2. CAIRN's general software development framework.

Figure 2 shows the global framework that is being developed in the group. Our compiler flow mixes several types of intermediate representations. The baseline representation is a simple tree-based model enriched with control flow information. This model is mainly used to support our source-to-source flow, and serves as the backbone for the infrastructure. We use the extensibility of the framework to provide more advanced representations along with their corresponding optimizations and code generation plug-ins. For example, for our pattern selection and accuracy estimation tools, we use a data dependence graph model in all basic blocks instead of the tree model. Similarly, to enable polyhedral based program transformations and analysis, we introduced a specific representation for affine control loops that we use to derive a Polyhedral Reduced Dependence Graph (PRDG). Our current flow assumes that the application is specified as a hierarchy of communicating tasks, where each task is expressed using C or Matlab/Scilab, and where the system-level representation and the target platform model are often defined using Domain Specific Languages (DSL).

Gecos (Generic Compiler Suite) is the main backbone of CAIRN's flow. It is an open source Eclipse-based flexible compiler infrastructure developed for fast prototyping of complex compiler passes. Gecos is a 100% Java based implementation and is based on modern software engineering practices such as Eclipse plugin or model-driven software engineering with EMF (Eclipse Modeling Framework). As of today, our flow offers the following features:

- An automatic floating-point to fixed-point conversion flow (for HLS and embedded processors). **ID.Fix** is an infrastructure for the automatic transformation of software code aiming at the conversion of floating-point data types into a fixed-point representation. http://idfix.gforge.inria.fr.
- A polyhedral-based loop transformation and parallelization engine (mostly targeted at HLS). http://gecos.gforge.inria.fr.
- A custom instruction extraction flow (for ASIP and dynamically reconfigurable architectures). **Durase** and **UPaK** are developed for the compilation and the synthesis targeting reconfigurable platforms and the automatic synthesis of application specific processor extensions. They use advanced technologies, such as graph matching and graph merging together with constraint programming methods.
- Several back-ends to enable the generation of VHDL for specialized or reconfigurable IPs, and SystemC for simulation purposes (e.g., fixed-point simulations).

6.2. Gecos

Participants: Steven Derrien [corresponding author], Nicolas Simon, Nicolas Estibals, Ali Hassan El-Moussawi.

Keywords: source-to-source compiler, model-driven software engineering, retargetable compilation.

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 100% 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 at http://gecos.gforge.inria.fr.

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 since 2012 it is used in the context of the ALMA European project. The Gecos infrastructure will also be used by the EMMTRIX start-up, a spin-off from tha ALMA project which aims at commercializing the results of the project.

Recent developments in Gecos have focused on polyhedral loop transformations and efficient SIMD code generation for fixed point arithmetic data-types as a part of the ALMA project. Significant efforts were also put to provide a coarse-grain parallelization engine targeting the data-flow actor model in the context of the COMPA ANR project.

6.3. ID.Fix: Infrastructure for the Design of Fixed-point Systems

Participants: Olivier Sentieys [corresponding author], Nicolas Simon.

Keywords: fixed-point arithmetic, source-to-source code transformation, accuracy optimization, dynamic range evaluation

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). The last developments allowed to have a complete compatibility with GeCos and to avoid the use of Matlab for LTI and recursive systems. In the context of the ANR DEFIS project, the ID.Fix tool has been reorganized to be integrated in the DEFIS toolflow.

6.4. PowWow: Power Optimized Hardware and Software FrameWork for Wireless Motes

Participants: Olivier Sentieys [corresponding author], Arnaud Carer.

Keywords: Wireless Sensor Networks, Low Power, Preamble Sampling MAC Protocol, Hardware and Software Platform

PowWow is an open-source hardware and software platform designed to handle wireless sensor network (WSN) protocols and related applications. Based on an optimized preamble sampling medium access (MAC) protocol, geographical routing and `protothread` library, PowWow requires a lighter hardware system than Zigbee [72] to be processed (memory usage including application is less than 10kb). Therefore, network lifetime is increased and price per node is significantly decreased.

CAIRN's hardware platform (see Figure 3) is composed of:

- The motherboard, designed to reduce power consumption of sensor nodes, embeds an MSP430 microcontroller and all needed components to process PowWow protocol except radio chip. JTAG, RS232, and I2C interfaces are available on this board.
- The radio chip daughter board is currently based on a TI CC2420.
- The coprocessing daughter board includes a low-power FPGA which allows for hardware acceleration for some PowWow features and also includes dynamic voltage scaling features to increase power efficiency. The current version of PowWow integrates an Actel IGLOO AGL250 FPGA and a programmable DC-DC converter. We have shown that gains in energy of up to 700 can be obtained by using FPGA acceleration on functions like CRC-32 or error detection with regards to a software implementation on the MSP430.
- Finally, a last daughter board is dedicated to energy harvesting techniques. Based on the energy management component LTC3108 from Linear Technologies, the board can be configured with several types of stored energy (batteries, micro-batteries, super-capacitors) and several types of energy sources (a small solar panel to recover photovoltaic energy, a piezoelectric sensor for mechanical energy and a Peltier thermal energy sensor).

PowWow distribution also includes a generic software architecture using event-driven programming and organized into protocol layers. The software is based on Contiki [84], and more precisely on the `Protothread` library which provides a sequential control flow without complex state machines or full multi-threading.

To optimize the network regarding a particular application and to define a global strategy to reduce energy, PowWow offers the following extra tools: over-the-air reprogramming, analytical power estimation based on software profiling and power measurements, a dedicated network analyzer to probe and fix transmissions errors in the network. More information can be found at http://powwow.gforge.inria.fr.

6.5. Ziggie: a Platform for Wireless Body Sensor Networks

Participants: Olivier Sentieys [corresponding author], Arnaud Carer.

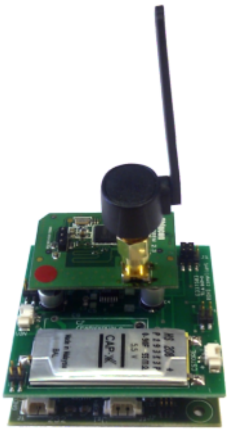


Figure 3. CAIRN*'s PowWow motherboard with radio and energy-harvesting boards connected*

Keywords: Wireless Body Sensor Networks, Low Power, Gesture Recognition, Localization, Hardware and Software Platform

The Zyggie sensor node has been developed in the team 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: an ATMEGA128RFA1 microcontroller, an MPU9150 Inertial Measurement Unit (IMU), an RF AS193 switch with two antennas, an LSP331AP barometer, a DC/DC voltage regulator with a battery charge controller, a wireless inductive battery charge controller, and some switches and control LEDs.

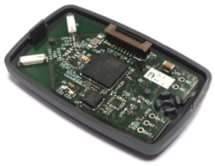


Figure 4. CAIRN*'s Ziggie platform for WBSN*

The IMU is composed of a 3-axis accelerometer, a 3-axis gyrometer and a 3-axis magnetometer. The IMU is communicating its data to the embedded microcontroller via an I2C protocol. We also developed our own MAC protocol for synchronization and data exchanges between nodes.

CAMUS Team

6. New Software and Platforms

6.1. APOLLO

Automatic speculative POLyhedral Loop Optimizer
FUNCTIONAL DESCRIPTION

We are developing a framework called APOLLO (Automatic speculative POLyhedral Loop Optimizer), 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 has been extended in 2015 to apply on-the-fly any kind of polyhedral transformations, including tiling, and to handle nonlinear loops as while-loops referencing memory through pointers and indirections.

- Participants: Aravind Sukumaran-Rajam, Juan Manuel Martinez Caamaño, Luis Esteban Campostrini, Artiom Baloian, Willy Wolff and Philippe Clauss
- Contact: Juan Manuel Martinez Caamaño

6.2. 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.

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

6.3. Clan

A Polyhedral Representation Extraction Tool for C-Based High Level Languages
FUNCTIONAL DESCRIPTION

Clan is a free software and library which translates some particular parts of high level programs written in C, 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++, 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/

6.4. 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/

6.5. 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.

- Participants: Imen Fassi, Philippe Clauss and Cédric Bastoul
- Contact: Philippe Clauss

6.6. XFOR-Wizard

XFOR-Wizard
FUNCTIONAL DESCRIPTION

Xfor-Wizard is a programming environment for XFOR programs, assisting users in writing XFOR codes and applying optimizing transformations. Automatic dependence analysis and comparisons against a referential code (XFOR-loops or classic for-loops) are achieved to order to help the user in ensuring semantic correctness of the written code.

- Participants: Imen Fassi, Philippe Clauss and Cédric Bastoul
- Contact: Philippe Clauss

6.7. XFORGEN

XFOR code generator
FUNCTIONAL DESCRIPTION

XFORGEN is a tool to automatically generate an XFOR code that is equivalent to for-loops that have been automatically transformed using a static polyhedral compiler. The generated XFOR code exhibits the parameters of the transformations that have been applied and thus can be modified for further optimizations.

- Participants: Imen Fassi, Philippe Clauss and Cédric Bastoul
- Contact: Philippe Clauss

6.8. 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/

6.9. ORWL and P99

ORWL is a reference implementation of the Ordered Read-Write Lock tools as described in [5]. The macro definitions and tools for programming in C99 that have been implemented for ORWL have been separated out into a toolbox called P99. ORWL is intended to become opensource, once it will be in a publishable state. P99 is available under a QPL at http://p99.gforge.inria.fr/.
Software classification: A-3-up, SO-4, SM-3, EM-3, SDL (P99: 4, ORWL: 2-up), DA-4, CD-4, MS-3, TPM-4

- Participants: Jens Gustedt, Mariem Saied, Daniel Salas
- Contact: Jens Gustedt
- http://p99.gforge.inria.fr/, http://orwl.gforge.inria.fr/

6.10. stdatomic and musl

We implement the libary side of the C11 atomic interface. It needs compiler support for the individual atomic operations and provides library supports for the cases where no low-level atomic instruction is available and a lock must be taken.

- This implementation builds entirely on the ABIs of the gcc compiler for atomics.
- It provide all function interfaces that the gcc ABIs and the C standard need.
- For compilers that don't offer the direct language support for atomics it provides a syntactically reduced but fully functional approach to atomic operations.
- At the core of the library is a new and very efficient futex-based lock algorithm that is implemented for the Linux operating system.

A description of the new lock algorithm has been given in [24]. A short version of it has been accepted for SAC'16.

The primary target of this library is an integration into musl to which we also contribute. It is 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
- http://stdatomic.gforge.inria.fr/, http://www.musl-libc.org/

6.11. PolyLib

The Polyhedral Library
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/

COMPSYS Project-Team

6. New Software and Platforms

6.1. Aspic

Accelerated Symbolic Polyhedral Invariant Generation
KEYWORDS: Abstract Interpretation - Invariant Generation
FUNCTIONAL DESCRIPTION

Aspic is an invariant generator for general counter automata. Combined with C2fsm (a tool developed by P. Feautrier in Compsys), it can be used to derive invariants for numerical C programs, and also to prove safety. It is also part of the WTC toolsuite (see http://compsys-tools.ens-lyon.fr/wtc/index.html), a tool chain to compute worse-case time complexity of a given sequential program.

Aspic implements the theoretical results of Laure Gonnord's PhD thesis on acceleration techniques and has been maintained since 2007.

- Participant: Laure Gonnord
- Contact: Laure Gonnord
- URL: http://laure.gonnord.org/pro/aspic/aspic.html

6.2. DCC

DPN C Compiler
KEYWORDS: Polyhedral compilation - Automatic parallelization - High-level synthesis
FUNCTIONAL DESCRIPTION

Dcc (Data-aware process network C compiler) analyzes a sequential regular program written in C and generates an equivalent architecture of parallel computer as a communicating process network (Data-aware Process Network, DPN). Internal communications (channels) and external communications (external memory) are automatically handled while fitting optimally the characteristics of the global memory (latency and throughput). The parallelism can be tuned. Dcc has been registered at the APP ("Agence de protection des programmes") and transferred to the XtremLogic start-up under an Inria license.

- Participants: Christophe Alias and Alexandru Plesco
- Contact: Christophe Alias

6.3. Lattifold

Lattice-based Memory Folding
KEYWORDS: Polyhedral compilation - Euclidean Lattices
FUNCTIONAL DESCRIPTION

Implements advanced lattice-based memory folding techniques. The idea is to reduce memory footprint of multidimensional arrays by reducing the size of each dimension. Given a relation denoting conflicting array cells, it produces a new mapping based on affine functions bounded by moduli. The moduli induces memory reuse and bound memory accesses to a tighter area, allowing to reduce the array size without loss of correctness.

- Partner: ENS Lyon
- Contact: Alexandre Isoard

6.4. OpenOrdo

OpenStream scheduler

FUNCTIONAL DESCRIPTION

Finding polynomial schedules for the streaming language OpenStream. Main use: detecting deadlocks.

- Contact: Paul Feautrier

6.5. PoCo

Polyhedral Compilation library
KEYWORDS: Polyhedral compilation - Automatic parallelization
FUNCTIONAL DESCRIPTION

PoCo (Polyhedral Compilation library) is a compilation framework allowing to develop parallelizing compilers for regular programs. PoCo features many state-of-the-art polyhedral program analysis (dependences, affine scheduling, code generation) and a symbolic calculator on execution traces (represented as convex polyhedra). PoCo has been registered at the APP ("agence de protection des programmes") and transferred to the XtremLogic start-up under an Inria license.

- Participant: Christophe Alias
- Contact: Christophe Alias

6.6. PolyOrdo

Polynomial Scheduler
FUNCTIONAL DESCRIPTION

Computes a polynomial schedule for a sequential polyhedral program having no affine schedule. Uses algorithms for finding positive polynomials in semi-algebraic sets. Status: proof of concept software.

- Contact: Paul Feautrier

6.7. PPCG-ParamTiling

Parametric Tiling Extension for PPCG
KEYWORDS: Source-to-source compiler - Polyhedral compilation
FUNCTIONAL DESCRIPTION

PPCG is a source-to-source compiler, based on polyhedral techniques, targeting GPU architectures. It involves automatic parallelization and tiling using polyhedral techniques. This version replaces the static tiling of PPCG by a fully parametric tiling and code generator. It allows to choose tile sizes at run time when the memory size is known. It also provides a symbolic expression of memory usage depending on the problem size and the tile sizes.

- Partner: ENS Lyon
- Contact: Alexandre Isoard

6.8. Termite

Termination of C programs
KEYWORDS: Abstract Interpretation - Termination
FUNCTIONAL DESCRIPTION

TERMITE is the implementation of our new algorithm "Counter-example based generation of ranking functions" (see Section 7.4). Based on LLVM and Pagai (a tool that generates invariants), the tool automatically generates a ranking function for each *head of loop*.

TERMITE represents 3000 lines of OCaml and is now available via the opam installer.

- Participants: Laure Gonnord, Gabriel Radanne (PPS, Univ Paris 7), David Monniaux (CNRS/Verimag).
- Contact: Laure Gonnord
- URL: https://termite-analyser.github.io/

6.9. Vaphor

Validation of C programs with arrays with Horn Clauses

KEYWORDS: Abstract Interpretation - Safety - Array Programs
FUNCTIONAL DESCRIPTION

VAPHOR (Validation of Programs with Horn Clauses) is the implementation of our new algorithm “An encoding of array verification problems into array-free Horn clauses” (see Section 7.3). The tool implements a translation from a C-like imperative language into Horn clauses in the SMT-lib Format.

VAPHOR represents 2000 lines of OCaml and its development is under consolidation.

- Participants: Laure Gonnord, David Monniaux (CNRS/Verimag).
- Contact: Laure Gonnord
- URL: not yet published, under consolidation.

CORSE Team

5. New Software and Platforms

5.1. Tirex

TIREX is an extensible, textual intermediate code representation that is intended to be used as an exchange format for compilers and other tools working on low level code. In the scope of the TIREX project we have developed tools for generating TIREX code from higher level languages such as C, as well as a number of static analyses and transformations.

Work on the TIREX project consisted of two main parts, firstly the cleanup and maintenance of the existing tools and web site and, secondly, implementing new backends for emitting TIREX.

The existing TIREX transormation and analysis tools as well the web site have been updated to make sure they work with the newest versions of their respective platforms (Java and PHP). They have also been refactored to make better use of newer or safer APIs. This work also included a redesign of the web site of the TIREX project and a rewrite of the build system.

The existing Open64 based backend has been updated to comply with the TIREX v2 specification so its output can be used with the rest of the tool chain.

We have also developed two new backends allowing us to generate TIREX code from any language the LLVM frontends support (including C, C++ and LLVM IR) as well as directly from assembly code. Preliminary work for generating TIREX directly from binaries has also been done, and the assembly backend is designed to allow most of its code to be reused for this purpose. These new developments required a partial rewrite of LLVMs internal machine description system to expose more machine information in an easily accessible manner. As a positive side effect we were able to reuse several parts used in earlier stages of the LLVM pipeline to write a simple type analysis on machine code used in the assembly backend. We also implemented a control flow reconstruction pass in the assembly backend to improve the quality of the generated code.

Lastly we have adopted continuous integration and started curating a regression test suite for our new developments.

5.2. LLVM plugins

Work has been started on multiple plugins for the LLVM compiler framework that implement the code optimization that have been elaborated by the team. While being work in progress this already provides us with crucial information for program analysis such as data-dependencies.

- Polly pointer disambiguation (publicly available): Status: Published. Description: A llvm-Polly patch that generates versioned SCoP, where the optimized version is guarded by run-time tests to validate that there are no hazardous aliasing.
- More on pointer disambiguation (to STMicroelectronics): Status: Implemented. Allows the use of malloc identifiers to quickly evaluate possible aliasing at run-time.
- Dynamic-dependence graph (to STMicroelectronics): Status: Under development. The run-time process is close to completion. Requires to treat function calls as sub-loops to allow optimization of recursive functions. The static analysis is capable of reading the trace file. The next step is to use a memory model to identify code transformations that would have better memory locality.

5.3. The klang-omp OpenMP compiler

Klang-Omp is a C and C++ source-to-source OpenMP compiler based on LLVM framework and on Intel's Clang-OMP front-end. It translates OpenMP directives into calls to task-based runtime system APIs. Klang-Omp currently targets both the StarPU runtime and the Kaapi runtime. The compiler supports independent tasks as defined by the 3.1 revision of the OpenMP specification as well as dependent tasks introduced with OpenMP 4. It also has been extended to support the omp target construct, making OpenMP applications able to offload computation to accelerators. This support also relies on the StarPU and XKaapi accelerator support capabilities. This work has been funded by the KSTAR Inria ADT project, involving the AVALON, STORM, MOAIS and CORSE Inria team. While the KSTAR project will end in January 2016, the klang-omp compiler will still be maintained and extended to support future OpenMP-oriented research actions, such as the ones promoted by the HEAVEN Persyval project.

5.4. mcGDB: Debugging of Multithreaded Applications

mcGDB is a new debugger for multithreaded applications. It implements a novel approach for interactive debugging named Programming Model-Centric Debugging. mcGDB raises interactive debugging to the level of programming models, by capturing and interpreting events generated during the application execution (e.g. through breakpointed API function calls). This new approach debugging is applied to four different programming models: software components (ST/NPM), Data flow (ST/PEDF), OpenCL and OpenMP. MCGDB was initialy developped by Kevin Pouget with STMicromectronics (CIFRE thesis). mcGDG uses the Temanejo graphical interface to display task graphs. mcGDB is currently extended in the DEMA/Nano2017 project with ST Microelectronics, Inria/Parkas and UPMC.

5.5. BOAST: Metaprogramming of Computing Kernels

BOAST aims at providing a framework to metaprogram, benchmark and validate computing kernels. BOAST is a programming framework dedicated to code generation and autotuning. This software allows the transformation from code written in the BOAST DSL to classical HPC targets like FORTRAN, C, OpenMP, OpenCL or CUDA. It also enables the meta-programming of optimization that can be (de)activated when needed. BOAST can also benchmark and do non regression tests on the generated kernels. This approach gives, both, performance gains and improved performance portability.

BOAST was used to generate and optimize the computing kernels of two scientific applications:

- BigDFT
- SPECFEM

BOAST can be dowloaded at this address https://forge.imag.fr/projects/boast/.

DREAMPAL Project-Team

5. New Software and Platforms

5.1. HoMade

KEYWORDS: SoC - Multicore - Softcore

FUNCTIONAL DESCRIPTION

HoMade is a softcore processor. The current version is reflective (i.e., the program it executes is self-modifiable), and statically configurable, dynamically reconfigurable multi-processors are the next steps. Users have to add to it the functionality they need in their applications via IPs. We have also being developing a library of IPs for the most common processor functions (ALU, registers, ...). All the design is in VHDL except for some schematic specifications.

- Participant: Jean Luc Dekeyser
- Partner: LIFL
- Contact: Jean Luc Dekeyser
- URL: https://sites.google.com/site/homadeguideen/home

5.2. JHomade

FUNCTIONAL DESCRIPTION

JHomade is a software suite written in JAVA, including compilers and tools for the HoMade processor. It allows us to compile HiHope programs to Homade machine code and load the resulting binaries on FPGA boards. It was first released in 2013. The second version in 2014 includes several new features, like a C-frontend, a few optimizations (automatic inlining and more compact byte-code), a binary decoder and a code-generator for VHDL simulation. New features of the HiHope language are described in [19].

- Contact: Frédéric Guyomarch
- URL: https://gforge.inria.fr/frs/?group_id=3646

POSTALE Team

5. New Software and Platforms

5.1. Boost.SIMD

FUNCTIONAL DESCRIPTION

Boost.SIMD provides a portable way to vectorize computation on Altivec, SSE or AVX while providing a generic way to extend the set of supported functions and hardwares.

- Contact: Joël Falcou
- URL: http://www.github.com/MetaScale/nt2

5.2. CovTrack

FUNCTIONAL DESCRIPTION

CovTrack: agile realtime multi-target tracking algorithm.

- Contact: Lionel Lacassagne

5.3. Dohko

FUNCTIONAL DESCRIPTION

Dohko is a goal-oriented cloud architecture that aims to simplify the cloud for the users through a declarative strategy. It implements the autonomic properties: self-configuration, self-healing, and context-awareness. In Dohko, the users specify the applications and the requirements (e.g., number of CPU cores, maximal financial cost per hour, among others), and the system automatically (a) selects the resources (i.e., VMs) that meet the constraints, (b) configures and installs the applications in the clouds, (c) handles resource failures, and (d) executes the applications.

- Contact: Alessandro Ferreira Leite
- URL: http://dohko.io/

5.4. Molly

FUNCTIONAL DESCRIPTION

Using Polly extension, the LLVM compiler framework is able to automatically parallelize general programs for shared memory threading for by exploiting the powerful analysis and transformations of the polyhedral model.

Molly adds the ability to manage distributed memory using the polyhedral model and is therefore able to automatically parallelize even for the largest of today's supercomputer. Once the distribution of data between the computer's nodes is known, Molly determines the values that are required to be transferred between the nodes and chunks them into as few messages as possible. It also keeps tracks of the buffers required by the MPI interface. Transfers are asynchronous such that further computations take place while the data is being transferred.

- Contact: Michael Kruse

5.5. MyNRC

FUNCTIONAL DESCRIPTION

MyNRC is multi-plateform library that can handle SSE, AVX, Neon and ST VECx registers.

- Contact: Lionel Lacassagne

5.6. NT2

Numerical Template Toolbox
FUNCTIONAL DESCRIPTION

The Numerical Template Toolbox (NT2) is an Open Source C++ library aimed at simplifying the development, debugging and optimization of high-performance computing applications by providing a Matlab like syntax that eases the transition between prototype and actual application.

- Participants: Joël Falcou, Pierre Estérie and Ian Masliah
- Contact: Joël Falcou
- URL: https://github.com/jfalcou/nt2

TASC Project-Team

6. New Software and Platforms

6.1. AIUR

(Artificial Intelligence Using Randomness)
FUNCTIONAL DESCRIPTION

The main idea is to be unpredictable by making some stochastic choices. The AI starts a game with a "mood" randomly picked up among 5 moods, dictating some behaviours (aggressive, fast expand, macro-game, ...). In addition, some other choices (productions, timing attacks, early aggressions, ...) are also taken under random conditions.

Learning is an essential part of AIUR . For this, it uses persistent I/O files system to record which moods are efficient against a given opponent, in order to modify the probability distribution for the mood selection. The current system allows both on-line and off-line learning.

- Contact: Florian Richoux
- URL: https://github.com/AIUR-group/AIUR

6.2. CHOCO

SCIENTIFIC DESCRIPTION

For fourth consecutive year, CHOCO has participated at the MiniZinc Challenge , an annual competition of constraint programming solvers. Since then, in concurrency with 16 other solvers, CHOCO has won two silver medals and four bronze medals in three out of four categories (Free search, Parallel search and Open class). Five versions have been released all year long, the last one (v3.3.3, Dec. 22th) has the particularity to be promoted on Maven Central Repository. The major modifications were related to an improvement of the overall solver (efficiency, stability and robustness) but also a simplification of the API. As an example, more flexibility has been injected to the search loop, a central concept of the solver.
FUNCTIONAL DESCRIPTION

CHOCO is a Free and Open-Source Software dedicated to Constraint Programming. It is a Java library written under BSD 4-clause license (700 classes, 134K lines of code). It aims at describing hard combinatorial problems in the form of Constraint Satisfaction Problems and solving them with Constraint Programming techniques. The user models its problem in a declarative way by stating the set of constraints that need to be satisfied in every solution. Then, CHOCO solves the problem by alternating constraint filtering algorithms with a search mechanism. In addition to native explanations system, soft constraints and global constraints, the library is, in practice, open, easy to integrate and to tweak. A User Guide is now available: 164 pages describing how to use CHOCO, together with responsive online support (forums and mailing-lists).

- Participants: Charles Prud'homme, Nicolas Beldiceanu, Jean-Guillaume Fages, Xavier Lorca, Thierry Petit and Rémi Douence
- Partner: Ecole des Mines de Nantes
- Contact: Charles Prud'homme
- URL: http://www.choco-solver.org/

6.3. GCCat

Global Constraint Catalog
KEYWORDS: Constraint Programming - Global constraint - Catalogue - Graph - Automaton - Transducer - First order formula - meta-data - ontology - symmetry - counting -

FUNCTIONAL DESCRIPTION

This global constraint catalog presents a catalogue of global constraints where each constraint is explicitly described in terms of graph properties and/or automata and/or first order logical formulae with arithmetic. When available, it also presents some typical usage as well as some pointers to existing filtering algorithms. This year we were preparing a second volume of the catalog focused on time-series constraints. It presents a restricted set of finite transducers used to synthesise structural time-series constraints described by means of a multi-layered functions composition scheme. Second it provides the corresponding synthesised catalogue of structural time-series constraints where each constraint is explicitly described in terms of automata with accumulators.

- Participants: Nicolas Beldiceanu, Mats Carlsson, Sophie Demassey and Helmut Simonis
- Contact: Nicolas Beldiceanu
- URL: http://sofdem.github.io/gccat/gccat/index.html

6.4. GHOST

General meta-Heuristic Optimization Solving Tool
FUNCTIONAL DESCRIPTION

GHOST, i.e. General meta-Heuristic Optimization Solving Tool, is a template C++ library designed for StarCraft:BroodWartm. GHOST implements a meta-heuristic solver aiming to solve any kind of combinatorial and optimization RTS-related problems represented by a csp /cop. The solver handles dedicated geometric and assignment constraints in a way that is compatible with very strong real time requirements.

- Contact: Florian Richoux
- URL: http://github.com/richoux/GHOST

6.5. IBEX

pour le calcul ensembliste (calcul numérique garanti avec propagation rigoureuse d'incertitudes)
KEYWORD: Constraint Programming
SCIENTIFIC DESCRIPTION

In 2014 the development on IBEX has focused on the following points:

Rejection test based on first-order conditions (see First Order Rejection Tests For Multiple-Objective Optimization, A. Goldsztejn et al. [42]).

Q-intersection (see Q-intersection Algorithms for Constraint-Based Robust Parameter Estimation, C. Carbonnel et al., AAAI 2014)
FUNCTIONAL DESCRIPTION

IBEX is a C++ library for solving nonlinear constraints over real numbers. The main feature of Ibex is its ability to build solver/paver strategies declaratively through the contractor programming paradigm. It also comes with a black-box solver and a global optimizer.

- Participants: Ignacio Araya, Gilles Chabert, Bertrand Neveu, Ignacio Salas Donoso and Gilles Trombettoni
- Partners: ENSTA - Ecole des Ponts ParisTech
- Contact: Gilles Chabert
- URL: http://www.ibex-lib.org/

AOSTE Project-Team

6. New Software and Platforms

6.1. SynDEx

KEYWORDS: Embedded systems - Real time - Optimization - Distributed - 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

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

6.2. 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: Frédéric Mallet, and Julien Deantoni
- Contact: Frédéric Mallet
- URL: http://timesquare.inria.fr

6.3. Lopht

KEYWORDS: Real-time scheduling, compilation, ARINC 653, TTEthernet, Many-core, Network-on-chip

SCIENTIFIC DESCRIPTION

Lopht is an acronym for Logical to Physical Time Compiler. Lopht has been designed as an implementation of the AAA methodology. Like SynDEx, Lopht relies on off-line allocation and scheduling techniques to allow real-time implementation of dataflow synchronous specifications (e.g. Scade/Heptagon) onto multiprocessor systems. The main originality is that Lopht takes a compilation-like approach based on:

- Precise modeling of its implementation platforms. For this reason, Lopht targets novel, more complex architectures such as many-core chips and time-triggered embedded systems based on standards such as ARINC 653 and TTEthernet.
- Taking into account complex non-functional specifications covering real-time (release dates and deadlines possibly different from period, major time frame, end-to-end flow constraints), ARINC 653 partitioning, the possibility to preempt or not each task, and finally SynDEx-like allocation
- Tight integration of program analysis, scheduling, and optimization approaches coming from 3 research fields (real-time scheduling, compilation, and synchronous languages) to improve the efficiency of resulting implementations while ensuring functional correctness, the respect of non-functional requirements, and scalability.

FUNCTIONAL DESCRIPTION Lopht is a software tool similar in functioning to a compiler. It takes as input one file defining the functional and non-functional specification of a system (including a model of the execution platform and non-functional requirements). It automatically produces all files needed to build a running implementation (the C code for each processor cores and the configuration files).

- Participants: Dumitru Potop-Butucaru, Keryan Didier
- Contact: Dumitru Potop-Butucaru (dumitru.potop@inria.fr)

6.4. EVT Kopernic

KEYWORD: Embedded systems

EVT Kopernic provides a probabilistic worst case execution time estimation for a program on a processor. The tool takes a set of measurements (execution times of the program on the processor) as input and it provides a probability distribution. The first version released in 2015 is restricted to independent data and a second version has been obtained for dependent data during the last part of the year. A third version provides rules for obtaining the measurements is to be released in the first part of 2016.

- Participants: Liliana Cucu and Adriana Gogonel
- Contact: Liliana Cucu
- URL: Currently restricted distribution

6.5. SAS

Simulation and Analysis of Scheduling

SCIENTIFIC DESCRIPTION

The SAS (Simulation and Analysis of Scheduling) software allows the user to perform the schedulability analysis of periodic task systems in the monoprocessor case.

The main contribution of SAS, when compared to other commercial and academic softwares of the same kind, is that it takes into account the exact preemption cost between tasks during the schedulability analysis. Beside usual real-time constraints (precedence, strict periodicity, latency, etc.) and fixed-priority scheduling policies (Rate Monotonic, Deadline Monotonic, Audsley++, User priorities), SAS additionaly allows to select dynamic scheduling policy algorithms such as Earliest Deadline First (EDF). The resulting schedule is displayed as a typical Gantt chart with a transient and a permanent phase, or as a disk shape called "dameid", which clearly highlights the idle slots of the processor in the permanent phase.

FUNCTIONAL DESCRIPTION

The SAS software allows the user to perform the schedulability analysis of periodic task systems in the monoprocessor case.

- Participants: Daniel De Rauglaudre and Yves Sorel
- Contact: Yves Sorel
- URL: http://pauillac.inria.fr/~ddr/sas-dameid/

CONVECS Project-Team

5. New Software and Platforms

5.1. The CADP Toolbox

Participants: Hubert Garavel [correspondent], Frédéric Lang, Radu Mateescu, Wendelin Serwe.

We maintain and enhance CADP (*Construction and Analysis of Distributed Processes* – formerly known as *CAESAR/ALDEBARAN Development Package*) [1], a toolbox for protocols and distributed systems engineering [0]. In this toolbox, we develop and maintain the following tools:

- CAESAR.ADT [42] 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 [47], [46] 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 [43] 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,

[0] http://cadp.inria.fr

  - 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 [51], CTL [39], ACTL [41], etc.) to be defined in XTL.
- PBG (*Partitioned BCG Graph*) is a file format implementing the theoretical concept of *Partitioned LTS* [45] 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 [56] 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 [44] 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.

5.2. The PMC Partial Model Checker

Participants: Radu Mateescu, Frédéric Lang.

We develop a tool named PMC (*Partial Model Checker*, see § 6.4), which performs the compositional model checking of dataless MCL formulas on networks of communicating automata described in the EXP language.

PMC can be freely downloaded from the CONVECS Web site [0].

[0] http://convecs.inria.fr/software/pmc

HYCOMES Team

5. New Software and Platforms

5.1. Flipflop

Test & Flip Net Synthesis Tool for the Inference of Technical Procedure Models
FUNCTIONAL DESCRIPTION

Flipflop is a Test and Flip net 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.

- Contact: Benoît Caillaud
- URL: http://tinyurl.com/oql6f3y

5.2. 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/

5.3. 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
- URL: https://bitbucket.org/cpenet/tnf_cpp

MUTANT Project-Team

6. New Software and Platforms

6.1. Antescofo

Participants: Arshia Cont, Jean-Louis Giavitto, Philippe Cuvillier, José Echeveste.

FUNCTIONAL DESCRIPTION Antescofo is a modular polyphonic Score Following system as well as a Synchronous Programming language for musical composition. The module allows for automatic recognition of music score position and tempo from a realtime audio Stream coming from performer(s), making it possible to synchronize an instrumental performance with computer realized elements. The synchronous language within Antescofo allows flexible writing of time and interaction in computer music.

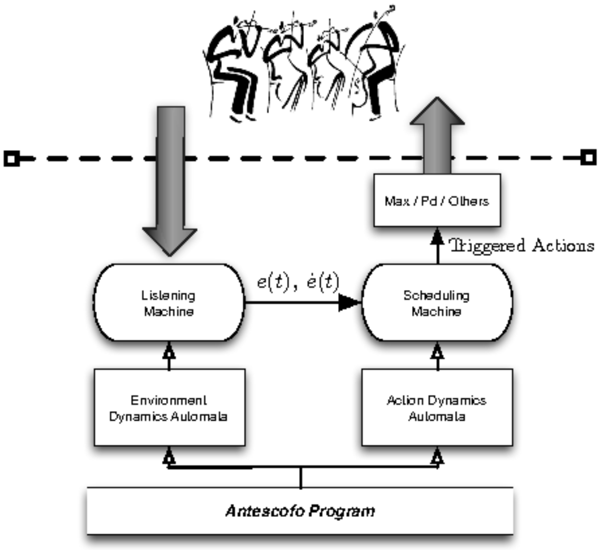


Figure 4. General scheme of Antescofo virtual machine

Antescofo v0.9 was released in November 2015. It contains major additions in the language (see Sections 7.5 and 7.6) as well as machine listning especially for singing voice and highly polyphonic instruments (See Release Notes). Antescofo Reference Guide is a collaborative document referencing the language and its usage, showcasing the software's latest developments.

- Participants: Arshia Cont, Jean-Louis Giavitto, Philippe Cuvillier and José Echeveste
- Contact: Arshia Cont
- URL: http://forumnet.ircam.fr/product/antescofo/

6.2. Ascograph

Participants: Arshia Cont, Grig Burloiu, Robert Piéchaud.

FUNCTIONAL DESCRIPTION

AscoGraph, the Antescofo graphical score editor released in 2013, provides a autonomous Integrated Development Environment (IDE) for the authoring of Antescofo scores. Antescofo listening machine, when going forward in the score during recognition, uses the message passing paradigm to perform tasks such as automatic accompaniment, spatialization, etc. The Antescofo score is a text file containing notes (chord, notes, trills, ...) to follow, synchronization strategies on how to trigger actions, and electronic actions (the reactive language). This editor shares the same score parsing routines with Antescofo core, so the validity of the score is checked on saving while editing in AscoGraph, with proper parsing errors handling. Graphically, the application is divided in two parts. On the left side, a graphical representation of the score, using a timeline with tracks view. On the right side, a text editor with syntax coloring of the score is displayed. Both views can be edited and are synchronized on saving. Special objects such as "curves", are graphically editable: they are used to provide high-level variable automation facilities like breakpoints functions (BPF) with more than 30 interpolations possible types between points, graphically editable.

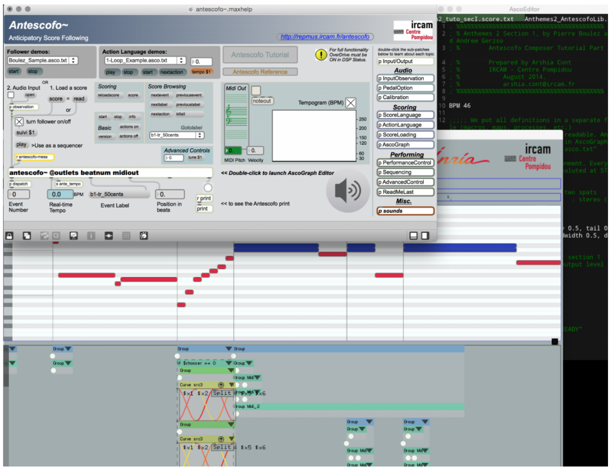


Figure 5. Antescofo and AscoGraph Screen Shorts (Nov. 2015)

In 2015, AscoGraph's User Interaction was redesigned as reported in [12], [13] and furthermore, a new Score Import procedure was developped and released in v0.25 (See Release Notes). See also 7.8 .

- Contact: Arshia Cont
- URL: http://forumnet.ircam.fr/product/antescofo/

6.3. Antescofo Timed Test Platform

Participants: Clément Poncelet, Florent Jacquemard, Pierre Donat-Bouillud.

The frequent use of Antescofo in live and public performances with human musicians implies strong requirements of temporal reliability and robustness to unforeseen errors in input. To address these requirements and help the development of the system and authoring of pieces by users, we are developing a platform for the automation of testing the behavior of Antescofo on a given score, with of focus on timed behavior. It is based on state of the art techniques and tools for *model-based testing* of embedded systems [37], and makes it possible to automate the following main tasks:

1. offline and on-the-fly generation of relevant input data for testing (i.e. fake performances of musicians, including timing values), with the sake of exhaustiveness,
2. computation of the corresponding expected output, according to a formal specification of the expected behavior of the system on a given mixed score,

3. black-box execution of the input test data on the System Under Test,
4. comparison of expected and real output and production of a test verdict.

The input and output data are timed traces (sequences of discrete events together with inter-event durations). Our method is based on formal models (specifications) in an ad hoc medium-level intermediate representation (IR). We have developed a compiler for producing automatically such IR models from Antescofo high level mixed scores.

Then, in the offline approach, the IR is passed, after conversion to Timed Automata, to the model-checker Uppaal, to which is delegated the above task (1), following coverage criteria, and the task (2), by simulation. In the online approach, tasks (1) and (2) are realized during the execution of the IR by a Virtual Machine developed on purpose. Moreover, we have implemented several tools for Tasks (3) and (4), corresponding to different boundaries for the implementation under test (black box): e.g. the interpreter of Antescofo's synchronous language alone, or with tempo detection, or the whole system.

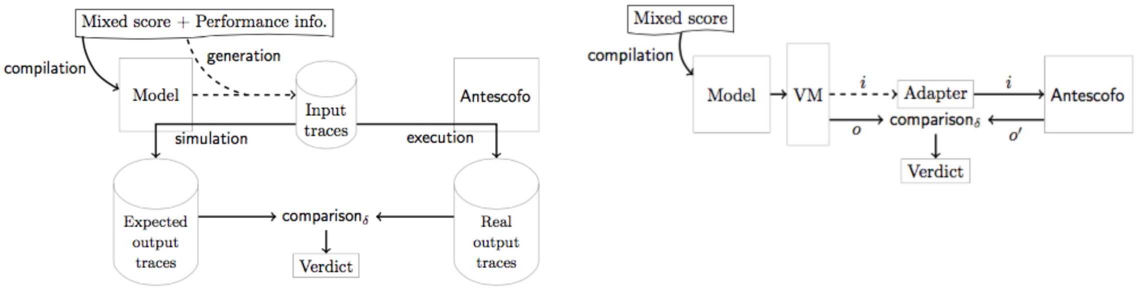


Figure 6. Offline and Online workflows for Antescofo Model Based Testing

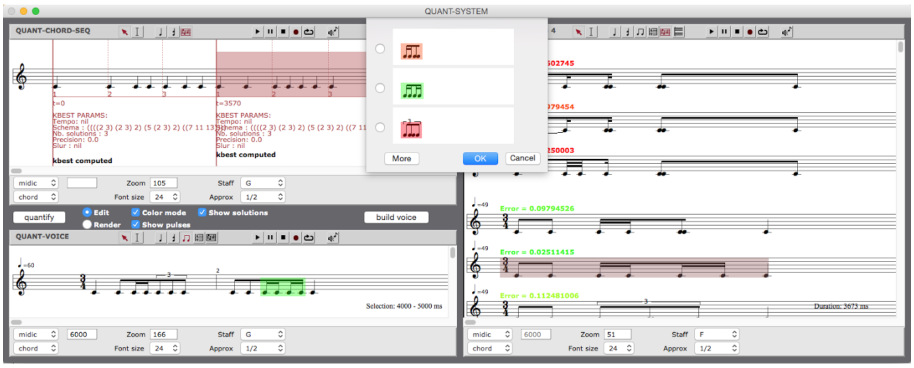
Our fully automatic framework has been applied to real mixed scores used in concerts and the results obtained have permitted to identify bugs in Antescofo.

6.4. Rhythm Quantization in Open Music

Participants: Adrien Ycart, Florent Jacquemard, Jean Bresson.

We are developing a new system for rhythm transcription, which is the conversion of sequences of timestamped discrete events into common-western music notation. The input events may e.g. come from a performance on a MIDI keyboard or may also be the result of a computation. Our system privileges the user interactions in order to search for a satisfying balances between different criteria, in particular the precision of the transcription and the readability of the music score in outcome. It is integrated in the graphical environment for computer assisted music composition OpenMusic, and and will be released publicly as a library of this system on the the Ircam's Forum.

We have developed a uniform approach for transcription, based on hierarchical representations of notation of duration as rhythm trees, and efficient algorithms for the lazy enumeration of solutions. It has been implemented via a dedicated interface making it possible the interactive exploration of the space of solutions, their visualization and their edition, with a particular focus on the processing of grace-notes and rests.



```
t=0
KBEST PARAMS:
Tempo: nil
Schema : ((((2 3) (2 3) 2) (5 (2 3) 2) ((7 11 13
Nb. solutions : 3
Precision: 0.0
Slur : nil
kbest computed
```

```
t=3570
KBEST PARAMS:
Tempo: nil
Schema : ((((2 3) (2 3) 2) (5 (2 3) 2) ((7 11
Nb. solutions : 3
Precision: 0.0
Slur : nil
kbest computed
```

● Edit ○ Render ☑ Color mode ☑ Show pulses ☑ Show solutions

midic chord Zoom 105 Font size 24 Staff G Approx 1/2

Selection: 4000 - 5000 ms

midic chord 6000 Zoom 166 Font size 24 Staff G Approx 1/2

502745

979454

250003

Error = 0.09794526

Error = 0.02511415

Error = 0.112481006

Duration: 3673 ms

midic chord 6000 Zoom 51 Font size 24 Staff F Approx 1/2

Figure 7. Rhythm Quantization in Open Music

PARKAS Project-Team

5. New Software and Platforms

5.1. Cmmtest: a tool for hunting concurrency compiler bugs

Participant: Francesco Zappa Nardelli [contact].

Languages, concurrency, memory models, C11/C++11, 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:

1. compiles the program using the compiler and compiler optimisations that are being tested;
2. runs the compiled program in an instrumented execution environment that logs all memory accesses to global variables and synchronisations;
3. 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.

Cmmtest is available from http://www.di.ens.fr/~zappa/projects/cmmtest/ and a list of bugs reported thanks to cmmtest is available from http://www.di.ens.fr/~zappa/projects/cmmtest/gcc-bugs.html.

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% free software, free in the sense that it respects the user's freedom.

The emphasis as now moved towards LLVM and its Polly framework for polyhedral compilation.

- Participants: Albert Cohen, Riyadh Baghdadi, Mircea Namolaru and Nhat Minh Le
- Contact: Albert Cohen
- URL: http://gcc.gnu.org/

5.3. Heptagon

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.

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

5.4. Ott and Lem

lightweight executable mathematics
FUNCTIONAL DESCRIPTION

Ott and Lem are lightweight tools for writing, managing, and publishing large scale semantic definitions, where the scale makes it hard to keep a definition internally consistent, and to keep a tight correspondence between a definition and implementations.

The two tools are complementary. Ott focuses on higher-level programming language semantics. Lem is domain-specific language that resembles a pure subset of Objective Caml, supporting typical functional programming constructs and common logical mechanisms. Both tools can generate OCaml, HOL4, Coq, and Isabelle code. They also generate LaTeX code for inclusion of the language definition in scientific documents. Ott also supports a Lem backend for a tight integration between the two tools.

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

5.5. 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.

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

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

5.7. 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: Sven Verdoolaege, Tobias Grosser, Michael Kruse, Chandan Reddy, Riyadh Baghdadi and Albert Cohen
- Contact: Sven Verdoolaege
- URL: http://repo.or.cz/w/ppcg.git

5.8. 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: Guillaume Baudart, in collaboration with Louis Mandel now at IBM Research
- Contact: Guillaume Baudart
- URL: http://rml.lri.fr

5.9. SundialsML

Sundials/ML
KEYWORDS: Simulation - Mathematics - Numerical simulations
FUNCTIONAL DESCRIPTION

Sundials/ML is a comprehensive OCaml interface to the Sundials suite of numerical solvers (CVODE, CVODES, IDA, IDAS, KINSOL, ARKODE). 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.

OCaml versions of the standard examples usually have an overhead of about 50% compared to the original C versions, and almost never more than 100%.

NEW PROGRESS

The current version of Sundials/ML comprises about 37,000 lines of OCaml (plus 15,000 lines of api documentation) and 16,000 lines of C (plus 1600 lines of commentary). This year we worked on updating the interface to support Sundials 2.6.x. This involved adding support for a new solver (ARKODE), new modules for sparse matrices (SuperLU_MT and KLU), new nvectors (pthreads and OpenMP), and new linear solvers (SPFGMR and PCG), as well as treating several other new or modified features. This work is almost complete and will be released early in 2016. The technical developments required to interface OCaml with this library are explained in a report which has been sumitted as a deliverable in the MODRIO project: “D.4.2.16—OCaml interface to the Sundials suite of numerical solvers”. This text will be developed and submitted for journal publication in early 2016.

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

5.10. 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.

NEW PROGRESS

- Development and release of a comprehensive manual: http://zelus.di.ens.fr/man/
- Progress on the interaction of multiple numeric solvers (masters internship of V. Andreani).
- New causality analysis (detection of algebraic loops).
- Participants: Marc Pouzet and Timothy Bourke
- Contact: Marc Pouzet
- http://zelus.di.ens.fr

5.11. 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: Sven Verdoolaege, Michael Kruse and Albert Cohen
- Contact: Sven Verdoolaege
- URL: http://repo.or.cz/w/isl.git

5.12. LaTeX package: Checklistings

FUNCTIONAL DESCRIPTION

User manuals and papers about programming languages usually contain many code samples, often with accompanying compiler messages giving the types of declarations or error messages explaining why certain declarations are invalid.

The checklistings package augments the fancyvrb and listings packages for including source code in LaTeX documents with a way to pass the source code through a compiler and also include the resulting messages in the document. It also integrates with the HeVeA tool developed in the Gallium team: http://hevea.inria.fr.

The motivation is to check the code samples in a document for syntax and typing errors and to facilitate the inclusion of inferred types and compiler warnings or errors in a text. This package is intentionally very lightweight and unlike packages like python it is not intended for interacting with an interpretor or including the execution traces of code. While checklistings does not focus on a specific programming language, it is designed to work well with ML-like languages.

We developed this package to improve the quality of our papers and presentations on the Zélus programming language, but it is designed to be general purpose and also works, for instance, with OCaml programs.

- Participants: Timothy Bourke and Marc Pouzet
- Contact: Timothy Bourke
- URL: http://www.ctan.org/pkg/checklistings

POSET Team

6. New Software and Platforms

6.1. i-score

The *i-score* software, whose first definition dates back to 2005 [1], [6], aims at offering graphical views, aka *scores*, of interactive system. It has already been used to experiment various graphical user interface proposals to define time constraints between processes. It has been the subject of Jaime Arias's PhD [11]. Several formalizations of its semantics have been proposed [36], [20], [21], [19]. In 2016, especially within the ADT project “Tuilage”, *i-score* independent modules should be identified and integrated as possible GUIs for the *T-calculus*.

6.2. T-calculus

Sketched in [9], the *T-calculus* is a Domain Specific Language [0] to provide simple and robust high-level description mechanisms of reactive systems. It will offer a programmatic view of the tile modeling paradigm [3], [8]. Its definition has been refined a number of times (see e.g. [9], [7] and [30], [18]). A prototype implementation of its reactive kernel has eventually been achieved in Haskell on top of the Euterpea libraries by the end of 2015. Its consolidation and further developments are now scheduled for 2016, especially within the ADT project “Tuilage”. Graphical representation of tiles should give rise to a robust correspondance between programmatic and graphical representations of reactive systems.

[0] See [40] for an early note by Hudak about the notion of Domain Specific Language, and see [39], [42] for application of this notion is computer music.

SPADES Project-Team

5. New Software and Platforms

5.1. COSYMA: Controller synthesis using multi-scale abstractions

FUNCTIONAL DESCRIPTION

CoSyMA is a tool for automatic controller synthesis for incrementally stable switched systems based on multi-scale discrete abstractions. The tool accepts as input a switched system defined by differential equations indexed by a set of modes, time and space sampling parameters used to define an approximation of the continuous state-space, and a safety or a time-bounded reachability specification. CoSyMA computes and refines discrete abstractions of the state space so as to generate a controller, if one exists, for the system that enforces the specification.

- Authors: Antoine Girard, Gregor Gössler, and Sebti Mouelhi.
- Partner: LJK.
- Contact: Gregor Gössler.

5.2. LoCa: Logical Causality Analyzer

FUNCTIONAL DESCRIPTION

Based on an execution trace, the component specifications, and a required property P, LoCA analyzes the causes of a violation of P in a component-based system. LoCA currently supports causality analysis in BIP and networks of timed automata. The core analysis engine is implemented as an abstract class, such that support for other models of computation (MoC) can be added by instantiating the class with the basic operations of the MoC.

- Authors: Lacramioara Astefanoaei, Yoann Geoffroy, and Gregor Gössler.
- Contact: Gregor Gössler.

5.3. LDDL: Coq proofs of circuit transformations for fault-tolerance

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 μFP [87]. 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 (see Section 6.3.3). 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: Dmitry Burlyaev and Pascal Fradet.
- Contact: Pascal Fradet.
- URL: https://team.inria.fr/spades/fthwproofs

5.4. pyCPA_TWCA: A pyCPA plugin for computing deadline miss models

FUNCTIONAL DESCRIPTION

We are developing pyCPA_TWCA, a pyCPA plugin for Typical Worst-Case Analysis as described in Section 6.2.5 . pyCPA is an open-source Python implementation of Compositional Performance Analysis developed at TU Braunschweig, which allows in particular response-time analysis. pyCPA_TWCA is an extension of this tool that is co-developed by Sophie Quinton and Zain Hammadeh (TU Braunschweig). It allows in particular the computation of weakly-hard guarantees for real-time tasks, *i.e.*, the number of deadline misses out of a sequence of executions. So far, pyCPA_TWCA is restricted to uniprocessor systems of independent tasks, scheduled according to static priority scheduling. A public release is planned for 2016.

- Contact: Sophie Quinton.

TEA Project-Team

6. New Software and Platforms

6.1. The Eclipse project POP

Participants: Loïc Besnard, Thierry Gautier, Paul Le Guernic, Jean-Pierre Talpin.

The distribution of project POP [0] is 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. Project-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.

Technically, project POP is composed of the Polychrony toolset, under GPL license, and its Eclipse framework, under EPL license. SSME (Syntactic Signal-Meta under Eclipse), is the metamodel 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 metamodel 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 toolchain. 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 toolchain.

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.

6.2. The Polychrony toolset

Participants: Loïc Besnard, Thierry Gautier, Paul Le Guernic, Jean-Pierre Talpin.

The Polychrony toolset is an Open Source development environment for critical/embedded systems. It is based on Signal, a real-time polychronous dataflow 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 toolset 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,
- assemble heterogeneous predefined components (bottom-up with COTS),
- generate executable code for various architectures.

[0] *Polychrony on POLARSYS (POP)*, an Eclipse project in the POLARSYS Industry Working Group, 2013. https://www.POLARSYS.org/projects/POLARSYS.pop

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

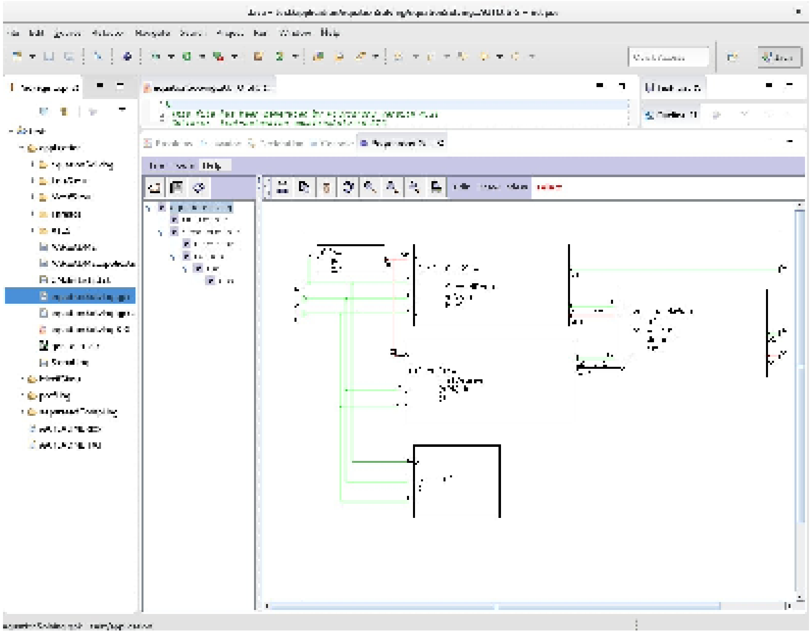


Figure 1. The Eclipse POP Environment

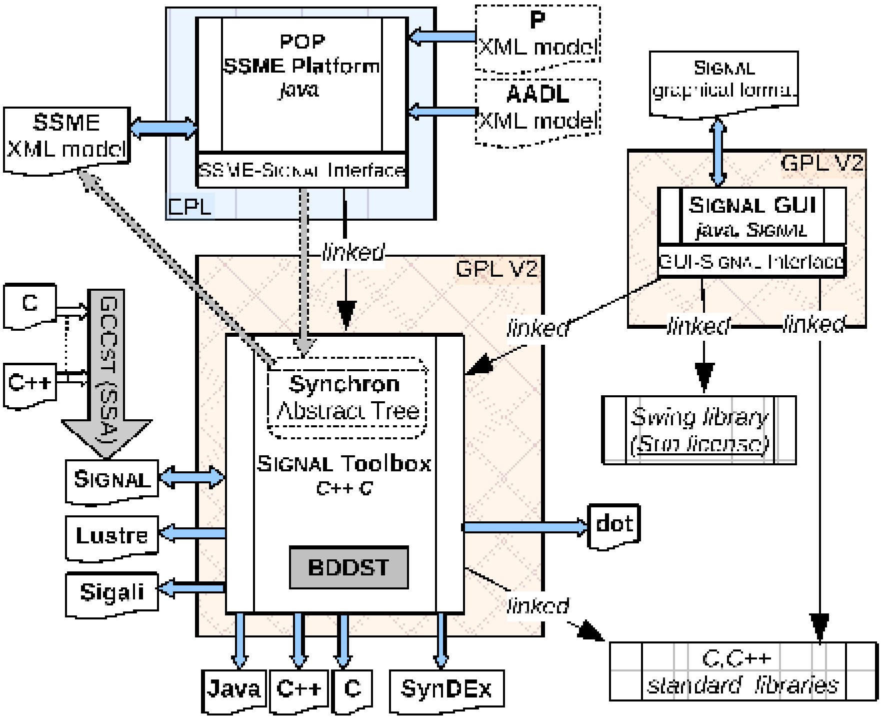


Figure 2. The Polychrony toolset high-level architecture

The Polychrony toolset 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. In 2015, 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 SSME platform, a front-end to the Signal toolbox in the Eclipse environment. It is distributed under EPL license.
- GCCst, a back-end to GCC that generates Signal programs (not yet available for download).

The Polychrony toolset also provides a large library of Signal programs and examples, user documentations and developer-oriented implementation documents, and facilities to generate new versions.

The Polychrony toolset can be freely downloaded on the following web sites:

- The Polychrony toolset public web site: http://polychrony.inria.fr/. This site, intended for users and for developers, contains downloadable executable and source versions of the software for differents platforms, user documentation, examples, libraries, scientific publications and implementation documentation. In particular, this is the site for the open-source distribution of Polychrony.
- The Inria GForge: https://gforge.inria.fr. This site, intended for internal developers, contains the whole sources of the environment and their documentation.

As part of its open-source release, the Polychrony toolset not only comprises source code libraries but also an important corpus of structured documentation, whose aim is not only to document each functionality and service, but also to help a potential developer to package a subset of these functionalities and services, and adapt them to developing a new application-specific tool: a new language front-end, a new back-end compiler. This multi-scale, multi-purpose documentation aims to provide different views of the software, from a high-level structural view to low-level descriptions of basic modules. It supports a distribution of the software "by apartment" (a functionality or a set of functionalities) intended for developers who would only be interested by part of the services of the toolset.

6.3. SigCert: translation validation from Signal to C

Participants: Van-Chan Ngo, Jean-Pierre Talpin, Thierry Gautier, Paul Le Guernic, Loïc Besnard.

Translation validation [00] is a technique that attempts to verify that program transformations preserve the program semantics. It is obvious to prove globally that the source program and its final compiled program have the same semantics. However, we believe that a better approach is to separate concerns and prove each analysis and transformation stage separately with respect to ad-hoc data-structures to carry the semantic information relevant to that phase.

In the case of the Signal compiler [1], [7], the preservation of the semantics can be decomposed into the preservation of clock semantics at the *clock calculation* phase [15] and that of data dependencies at the *static scheduling* phase[16], and, finally, value-equivalence of variables at the *code generation* phase[14].

Translation Validation for Clock Transformations in a Synchronous Compiler. The clock semantics of the source and transformed programs are formally represented as *clock models*. A clock model is a first-order logic formula that characterizes the presence/absence status of all signals in a Signal program at a given instant. Given two clock models, a *clock refinement* between them is defined which expresses the semantic preservation of clock semantics[15]. A method to check the existence of clock refinement is defined as a satisfiability problem which can be automatically and efficiently proved by a SMT solver [0].

[0] *Translation validation.* Pnueli A., Siegel M., and Singerman E. In Proceedings of TACAS'98, 1998.

[0] *Translation validation: From signal to c.* M. Siegel A. Pnueli and E. Singeman. In Correct Sytem Design Recent Insights and Advances, 2000.

[0] *Satisfiability modulo theories: An appetizer.* L. de Moura and N. Bjorner. In Brazilian Symposium on Formal Methods, 2009.

Precise Deadlock Detection for Polychronous Data-flow Specifications. Dependency graphs are a commonly used data structure to encode the streams of values in data-flow programs and play a central role in scheduling instructions during automated code generation from such specifications. We propose a precise and effective method that combines a structure of dependency graph and first order logic formulas to check whether multi-clocked data-flow specifications are deadlock-free before generating code from them. We represent the flow of values in the source programs by means of a dependency graph and attach first-order logic formulas to condition these dependencies. We use an SMT solver to effectively reason about the implied formulas and check deadlock freedom [16].

Implementation and Experiments. At a high level, our prototype tool *SigCert* ([14]) developed in OCaml could check the correctness of the compilation of Signal compiler w.r.t clock semantics, data dependence, and value-equivalence as given in Figure 3 . The individual modules designed in the context of this work are now being implemented and integrated in the open-source Polychrony toolset.

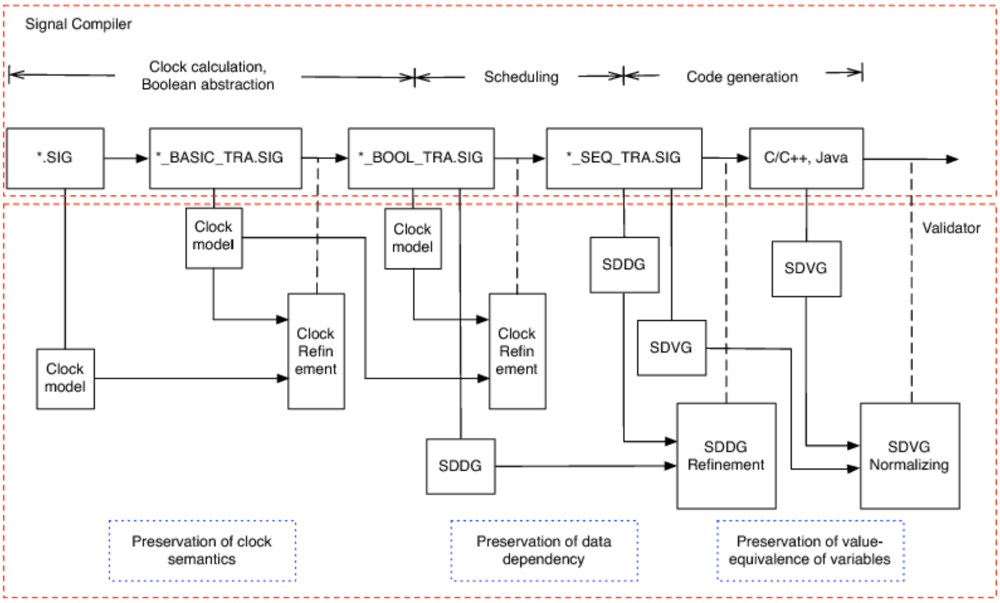


Figure 3. Our Integration within Polychrony Toolset

6.4. ADFG: Affine data-flow graphs scheduler synthesis under Eclipse

Participants: Alexandre Honorat, Jean-Pierre Talpin, Thierry Gautier, Loïc Besnard.

We have proposed a dataflow design model [2] of SCJ/L1 applications [0] in which handlers (periodic and aperiodic actors) communicate only through lock-free channels. Hence, each mission is modeled as a dataflow graph. The presented dataflow design model comes with a development tool integrated in the Eclipse IDE for easing the development of SCJ/L1 applications and enforcing the restrictions imposed by the design model. It consists of a GMF editor where applications are designed graphically and timing and buffering parameters can be synthesized. Indeed, abstract affine scheduling is first applied on the dataflow subgraph, 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 fixed-priority schedulability analysis (i.e., synthesis of timing and scheduling parameters of actors) considers both periodic and aperiodic actors.

[0] *Safety critical Java technology specification*. JSR-302, Year = 2010

Through a model-to-text transformation, using Acceleo, the SCJ code for missions, interfaces of handlers, and the mission sequencer is automatically generated in addition to the annotations needed by the memory checker. Channels are implemented as cyclic arrays or cyclical asynchronous buffers; and a fixed amount of memory is hence reused to store the infinite streams of tokens. The user must provide the SCJ code of all the `handleAsyncEvent()` methods. We have integrated the SCJ memory checker [0] in our tool so that potential dangling pointers can be highlighted at compile-time. To enhance functional determinism, we would like to develop an ownership type system to ensure that actors are strongly isolated and communicate only through buffers.

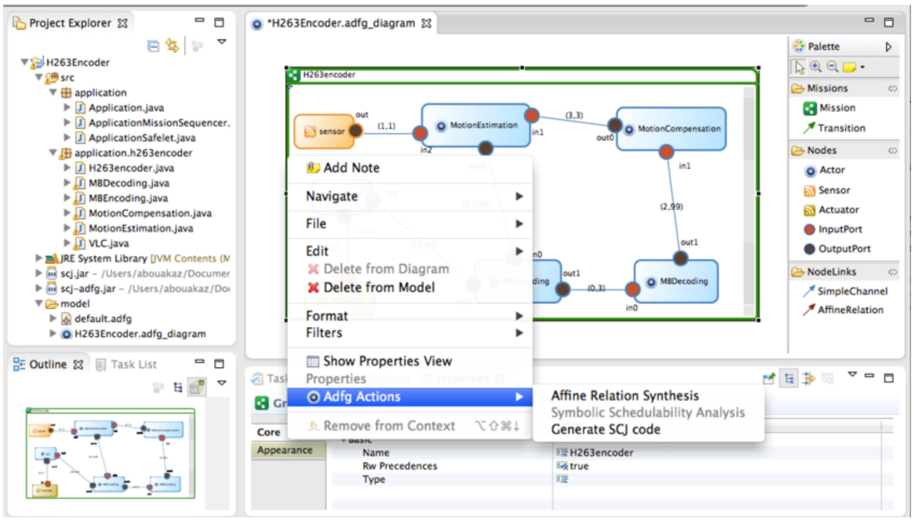


Figure 4. The ADFG Tool

The ADFG tool is being further developed in the context of the ADT "La vie d'AADL" in order to serve both as scheduler synthesis tool from AADL specifications and SCJ tasksets. We plan to further the front end analysis tools from Java task sets in order to build the input CSDF graphs from program analysis, in the context of a future PhD.

[0] *Static checking of safety critical Java annotations*. Tang, D. Plsek, A. and Vitek, J. International Workshop on Java Technologies for Real-Time and Embedded Systems, 2010

ANTIQUE Project-Team

5. New Software and Platforms

5.1. 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/

5.2. Astrée

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 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: Patrick Cousot, Jérôme Feret, Laurent Mauborgne, Antoine Miné and Xavier Rival
- Partner: CNRS
- Contact: Patrick Cousot
- URL: http://www.astree.ens.fr/

5.3. AstréeA

The AstréeA Static Analyzer of Asynchronous Software

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: Patrick Cousot, Radhia Cousot, Jérôme Feret, Antoine Miné and Xavier Rival est toujours membre de Inria. logiciels Inria): https://bil.inria.fr/
- Contact: Patrick Cousot
- URL: http://www.astreea.ens.fr/

5.4. ClangML

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: François Berenger, Pippijn Van Steenhoven and Devin Mccoughlin toujours membre de Inria. Inria): https://bil.inria.fr/
- Contact: François Berenger
- URL: https://github.com/Antique-team/clangml/tree/master/clang

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

5.6. 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

5.7. MemCAD

The MemCAD static analyzer
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 350 small size test cases that are used as regression tests.

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

5.8. OPENKAPPA

La platte-forme de modélisation OpenKappa
KEYWORDS: Systems Biology - Modeling - Static analysis - Simulation - Model reduction
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: Pierre Boutillier, Vincent Danos, Jérôme Feret, Walter Fontana, Russ Harmer, Jean Krivine and Kim Quyen Ly
- Partners: ENS Lyon - Université Paris-Diderot - Harvard Medical School
- Contact: Jérôme Feret
- URL: http://www.kappalanguage.org/

5.9. 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

5.10. Translation Validation

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 guara 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
- Contact: Xavier Rival

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

5.12. CELIA

The MemCAD static analyzer

FUNCTIONAL DESCRIPTION

CELIA is a tool for the static analysis and verification of C programs manipulating dynamic lists. The static analyzer computes for each control point of a C program the assertions which are true (i.e., invariant) at this control point. The specification language is a combination of Separation Logic with a first order logic over sequences of integers. The inferred properties describe the shape of the lists, their size, the relations between the data (or the sum, or the multiset of data) in list cells. The analysis is inter-procedural, i.e., the assertions computed relate the procedure local heap on entry to the corresponding local heap on exit of the procedure. The results of the analysis can provide insights about equivalence of procedures on lists or null pointer dereferencing. The analysis is currently extended to programs manipulating concurrent data structures.

- Participants: Ahmed Bouajjani, Cezara Drăgoi, Constantin Enea, Mihaela Sighireanu
- Contact: Cezara Drăgoi
- URL: http://www.liafa.jussieu.fr/celia/

CELTIQUE Project-Team

5. New Software and Platforms

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

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

5.3. 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: Frédéric Besson, David Pichardie and Laurent Guillo
- Contact: David Pichardie
- URL: http://sawja.inria.fr/

5.4. SAWJA

Static Analysis Workshop for Java
KEYWORDS: Security - Software - Code review
FUNCTIONAL 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.

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

5.5. 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)

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

5.6. CompCertSSA

KEYWORDS: Verified compilation - Single Static Assignment form - Optimization - Coq - OCaml

FUNCTIONAL DESCRIPTION

CompCertSSA is built on top of the C CompCert verified compiler, by adding a SSA-based middle-end (conversion to SSA, SSA-based optimizations, destruction of SSA). It is verified in the Coq proof assistant.

- Participant: Delphine Demange, David Pichardie, Yon Fernandez de Retana, Leo Stefanesco
- Contact: Delphine Demange
- URL: http://compcertssa.gforge.inria.fr/

DEDUCTEAM Team

6. New Software and Platforms

6.1. Introduction

Deducteam develops several kinds of tools or libraries:

- Proof checkers:
  - Dedukti: proof checker for the $\lambda\Pi$-calculus modulo rewriting
  - Sukerujo: extension of Dedukti with syntactic constructions for records, strings, lists, etc.
  - Rainbow: CPF termination certificate verifier
- Tools for translating into Dedukti's proof format proofs coming from various other provers:
  - Coqine translates Coq proofs
  - Focalide translates Focalize proofs
  - Holide translates OpenTheory proofs (HOL-Light, HOL4, ProofPower)
  - Krajono translates Matita proofs
  - Sigmaid translates ς-calculus
- Automated theorem provers:
  - iProverModulo: theorem prover based on polarized resolution modulo
  - SuperZenon: extension of Zenon using superdeduction
  - ZenonArith: extension of Zenon using the simplex algorithm for arithmetic
  - ZenonModulo: extension of Zenon using deduction modulo and producing Dedukti proofs
  - Zipperposition: superposition prover featuring arithmetic and induction
  - HOT: automated termination prover for higher-order rewrite systems
- Libraries or generation tools:
  - CoLoR: Coq library on rewriting theory and termination
  - Logtk: library for first-order automated reasoning
  - mSat: modular SAT/SMT solver with proof output
  - Moca: generator of construction functions for types with relations on constructors

In the following, we only details software that received improvements in 2015.

In addition, Shuai Wang developed the ProofCloud prototype, a proof retrieval engine for verified higher order proofs. ProofCloud provides a fast proof searching service for mathematicians and computer scientists for the reuse of proofs and proof packages. Using ProofCloud, he conducted a statistical analysis of the OpenTheory repository.

6.2. Autotheo

Autotheo is a tool that transforms axiomatic theories into polarized rewriting systems, thus making them usable in iProver Modulo. 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 iProver Modulo.

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

In 2015, we extended Autotheo so that it prints a derivation of the transformation of the axioms into rewriting rules. This derivation is in TSTP format and includes the CNF conversions obtained from the prover E.

6.3. CoLoR

CoLoR is 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.).

- Contact: Frédéric Blanqui
- URL: http://color.inria.fr/

In 2015, CoLoR has been enriched and improved in various ways:

- Its compilation time has been improved by about 20%.
- The results on computability have been extended to η-reduction.
- It has been enriched by a library on finite and infinite sets, and a proof of the infinite Ramsey's theorem [54].
- CoLoR is now available on OPAM.

6.4. Coqine

Coqine translates Coq proofs into Dedukti proofs.

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

The addition of higher-order pattern matching in Dedukti allowed the encoding of universes.

6.5. Dedukti

Dedukti is a proof-checker for the $\lambda\Pi$-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.

- Contact: Olivier Hermant
- URL: http://dedukti.gforge.inria.fr/

The new version of Dedukti (v2.5) brings two major improvements.

First the typing of rewrite rules has been completely reworked. It can now check a large class of rewrite rules including rules whose left-hand sides are not algebraic nor well-typed. Moreover the typing context do not need to be given with the rewrite rule anymore, as it is inferred by Dedukti, and therefore it is more convenient for the user.

Second, Dedukti can now be interfaced with automatic confluence checkers in order to check that the rewrite system generated by the rewrite rules together with beta reduction is confluent. This verification is important as the soundness of the program relies on this hypothesis.

6.6. Focalide

Focalide is an extension of the FoCaLize compiler which produces Dedukti files.

- Contact: Raphaël Cauderlier
- URL: http://deducteam.gforge.inria.fr/focalide/

Focalide has been improved to support FoCaLiZe proofs found by Zenon using the Dedukti backend for Zenon. This backend has been improved by a simple typing mechanism in order to work with Focalide. Focalide has also been updated again to work with the latest version of FoCaLiZe.

6.7. Holide

Holide translates HOL proofs to Dedukti proofs, using the OpenTheory standard (common to HOL Light and HOL4).

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

Shuai Wang fixed a number of problems, especially in the translation of type variables, allowing us to translate more libraries.

6.8. iProverModulo

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, a refinement of the resolution method based on deduction modulo. 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. iProver Modulo is also able to produce proofs that can be checked by Dedukti, therefore improving confidence.

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

In 2015, we improved its integration with Autotheo.

6.9. Krajono

Krajono translates Matita proofs into Dedukti proofs.

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

First working version able to translate the Matita library on arithmetics.

6.10. mSAT

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
- URL: https://github.com/Gbury/mSAT

mSAT now provides a functor for generating a McSat solver, outputs a model or a proof, and provides a push/pop functionality.

6.11. ZenonModulo

Zenon Modulo is an extension of the automated theorem prover Zenon. Compared to Super Zenon, it can deal with rewrite rules both over propositions and terms. Like Super Zenon, Zenon Modulo is able to deal with any first-order theory by means of a similar heuristic.

- Contact: Pierre Halmagrand
- URL: http://deducteam.gforge.inria.fr/zenonmodulo/

In 2015, we extended Zenon Modulo to polymorphism. Moreover, it can now take TPTP-TFF1 problems as input, and output Dedukti's proofs.

Guillaume Bury continued to improve an extension of Zenon with arithmetic.

6.12. Zipperposition

Zipperposition is an implementation of the superposition method that relies on the library Logtk for basic logic data structures and algorithms. Zipperposition is designed as a testbed for extensions to superposition, and can currently deal with polymorphic typed logic, integer arithmetic and total orderings.

- Contact: Simon Cruanes
- URL: http://deducteam.gforge.inria.fr/zipperposition/

In 2015, we extended Zipperposition to structural induction.

ESTASYS Team

5. New Software and Platforms

5.1. PLASMA Lab

Platform for Learning and Advanced Statistical Model checking Algorithms
KEYWORDS: Model Checking - Statistical - Model Checker - Runtime Analysis - Statistics
SCIENTIFIC DESCRIPTION

Statistical model checking (SMC) is a fast emerging technology for industrial scale verification and optimisation problems. Plasma was conceived to have high performance and be extensible, using a proprietary virtual machine. Since SMC requires only an executable semantics and is not constrained by decidability, we can easily implement different modelling languages and logics.
FUNCTIONAL DESCRIPTION

Plasma-Lab is a formal verification tool for complex embeded systems. It uses statistical model checking, and applies to complex problems coming from the area of security, cyber physical systems, or privacy.

- Participants: Axel Legay, Sean Sedwards, Louis-Marie Traonouez, Jean Quilbeuf
- Contact: Axel Legay
- URL: https://project.inria.fr/plasma-lab

5.2. PyECDAR

KEYWORDS: Timed input - Output automata
SCIENTIFIC DESCRIPTION

The tool has been originally developed to analyze the robustness of timed specifications, in extension of the tool Ecdar. As Ecdar , it allows to compose components specifications based on Timed I/O Automata (TIOA), and it implements timed game algorithms for checking consistency and compatibility. Additionally, it features original methods for checking the robustness of these specifications.

The tool has been later extended to analyse adaptive systems. It therefore implements original algorithms for checking featured timed games against requirements expressed in the timed AdaCTL logic.

The tool is written in Python with around 3’000 lines of code. It uses a Python console as user interface, from which it can load TIOA components from XML files written in the UPPAAL format, and design complex systems by combining the components using a simple algebra. Then, it can analyze these systems, transform them and save them in a new XML file.
FUNCTIONAL DESCRIPTION

PyEcdar is a free software that analyses timed games and timed specifications. The goal of the tool is to allow a fast prototyping of new analysis techniques. It currently allows to solve timed games based on timed automata models. These can be extended with adaptive features to represent dynamicity and to model software product lines.

- Participants: Louis-Marie Traonouez and Axel Legay
- Contact: Louis-Marie Traonouez
- URL: https://project.inria.fr/pyecdar/

5.3. Quail

FUNCTIONAL DESCRIPTION

Privacy is a central issue for Systems of Systems and interconnected objects. We propose QUAIL, a tool that can be used to quantify privacy of components. QUAIL is the only tool able to perform an arbitrary-precision quantitative analysis of the security of a system depending on private information. Thanks to its Markovian semantics model, QUAIL computes the correlation between the system's observable output and the private information, obtaining the amount of bits of the secret that the attacker will infer by observing the output.

- Participants: Fabrizio Biondi, Axel Legay, Louis-Marie Traonouez and Andrzej Wasowski
- Contact: Axel Legay
- URL: https://project.inria.fr/quail/

GALLIUM Project-Team

6. New Software and Platforms

6.1. CompCert

Participants: Xavier Leroy [**contact**], Sandrine Blazy [team Celtique], Jacques-Henri Jourdan, Bernhard Schommer [AbsInt GmbH].

The CompCert project investigates the formal verification of realistic compilers usable for critical embedded software. Such verified compilers come with a mathematical, machine-checked proof that the generated executable code behaves exactly as prescribed by the semantics of the source program. By ruling out the possibility of compiler-introduced bugs, verified compilers strengthen the guarantees that can be obtained by applying formal methods to source programs. AbsInt Angewandte Informatik GmbH sells a commercial version of CompCert with long-term maintenance.

- URL: http://compcert.inria.fr/ (academic), http://www.absint.com/compcert/ (commercial).

6.2. Diy

Participants: Luc Maranget [**contact**], Jade Alglave [Microsoft Research, Cambridge], Keryan Didier.

The **diy** suite (for “Do It Yourself”) 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, ARM or generic (LISA) 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. Recent versions also take a subset of the C language as input, so as to test and simulate the C11 model.

- URL: http://diy.inria.fr/

6.3. Menhir

Participants: François Pottier [**contact**], Yann Régis-Gianas [Université Paris Diderot].

Menhir is a LR(1) parser generator for the OCaml programming language. That is, Menhir compiles LR(1) grammar specifications down to OCaml code.

- URL: http://gallium.inria.fr/~fpottier/menhir/

6.4. OCaml

Participants: Damien Doligez [**contact**], Alain Frisch [LexiFi], Jacques Garrigue [Nagoya University], Fabrice Le Fessant, Xavier Leroy, Luc Maranget, Gabriel Scherer, Mark Shinwell [Jane Street], Leo White [Jane Street], Jeremy Yallop [OCaml Labs, Cambridge University].

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 SPARC), a debugger, a documentation generator, a compilation manager, a package manager, and many libraries contributed by the user community.

- URL: http://ocaml.org/

6.5. PASL

Participants: Mike Rainey [**contact**], Arthur Charguéraud, Umut Acar.

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.

- URL: http://deepsea.inria.fr/pasl/

6.6. Zenon

Participants: Damien Doligez [**contact**], Guillaume Bury [CNAM], David Delahaye [CNAM], Pierre Halmagrand [team DEDUCTEAM], Olivier Hermant [MINES ParisTech].

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 easily retargeted to output scripts for different frameworks (for example, Isabelle and Dedukti).

- URL: http://opam.ocaml.org/packages/zenon/zenon.0.8.0/

MARELLE Project-Team

5. New Software and Platforms

5.1. Coq

KEYWORDS: Proof - Certification - Formalisation
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, ...

- Participants: Benjamin Grégoire, Enrico Tassi, Bruno Barras, Yves Bertot, Pierre Boutillier, Xavier Clerc, Pierre Courtieu, Maxime Denes, Stéphane Glondu, Vincent Gross, Hugo Herbelin, Pierre Letouzey, Assia Mahboubi, Julien Narboux, Jean-Marc Notin, Christine Paulin-Mohring, Pierre-Marie Pédrot, Loïc Pottier, Matthias Puech, Yann Régis-Gianas, François Ripault, Matthieu Sozeau, Arnaud Spiwack, Pierre-Yves Strub, Benjamin Werner, Guillaume Melquiond and Jean-Christophe Filliâtre
- Partners: CNRS - Université Paris-Sud - ENS Lyon - Université Paris-Diderot
- Contact: Hugo Herbelin
- URL: http://coq.inria.fr/

Enrico Tassi and Maxime Dénès brought notable contributions to the Coq system in 2015. In particular, Enrico worked on the new user-interface that makes it possible to have several logical engines working on proofs simultaneously and Maxime Dénès supervised the release process for Coq 8.5, to be released in the early days of January.

In 2015, the Coq system is the object of intense activity within the Marelle project-team. Yves Bertot and Maxime Dénès are working at creating a consortium around this system, so that academic and industrial users find a suitable structure to voice there wishes for the evolution of the system, fund improvements, and coordinate developments for further improvement. This work is done in close collaboration with the $\pi.r^2$ project-team.

A first outcome of this animation work is the organization of regular events for developers to meet (coding sprints), the first of which happened in Sophia Antipolis in June 2015. Subsequently, Maxime Dénès was hired in Sophia Antipolis (in the Marelle project-team), and Matej Kosik was hired in Paris (in the $\pi.r^2$) team. A close collaboration was also set up with the Massachusetts Institute of Technology (MIT), with a software engineer to be hired at MIT to work on Coq in early 2016.

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

5.3. 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.

- Participants: Andrea Asperti, Jeremy Avigad, Yves Bertot, Cyril Cohen, François Garillot, Georges Gonthier, Stéphane Le Roux, Assia Mahboubi, Sidi Ould Biha, Ioana Pasca, Laurence Rideau, Alexey Solovyev, Enrico Tassi and Russell O'connor
- Contact: Assia Mahboubi
- URL: http://www.msr-inria.fr/projects/mathematical-components-2/

5.4. Ssreflect

FUNCTIONAL DESCRIPTION

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

- Participants: Cyril Cohen, Yves Bertot, Laurence Rideau, Enrico Tassi and Laurent Théry
- Contact: Yves Bertot
- URL: http://ssr.msr-inria.inria.fr/

5.5. Zoocrypt

FUNCTIONAL DESCRIPTION

ZooCrypt 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://www.easycrypt.info/zoocrypt/

MEXICO Project-Team

6. New Software and Platforms

6.1. General Remark

The team's software and platform are the same as in 2014, namely

- COSMOS,
- MOLE,
- CosyVerif;

no major changes have occurred in 2015.

PARSIFAL Project-Team

6. New Software and Platforms

6.1. 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.

In 2015, Abella has been extended with

- support for polymorphic definitions and theorems;
- schemas and automatically derived theorems about them;
- the ability to record and replay automated search;
- a number of new examples from process calculi, including a contributed example from Horace Blanc about relating the π-calculus and the λ-calculus.

One further development is that Abella can now be compiled into JavaScript and run completely inside any modern browser, thanks to the `js_of_ocaml` compiler from OCaml bytecode to JavaScript. We expect this to become rather crucial in popularization of Abella, particularly in a pedagogical context, since it does not require any local software installation—just a modern web browser.

- Participants: Dale Miller, Kaustuv Chaudhuri, Horace Blanc
- Partner: Department of Computer Science and Engineering, University of Minnesota
- Contact: Kaustuv Chaudhuri
- URL: http://abella-prover.org/
- Online version: http://abella-prover.org/tutorial/try

6.2. 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 expression possibly containing 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.

- Participant: Roberto Blanco Martinez
- Contact: Quentin Heath
- URL: http://slimmer.gforge.inria.fr/bedwyr/

6.3. 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: Tomer Libal and Giselle Reis
- Contact: Tomer Libal
- URL: https://github.com/proofcert/checkers

6.4. Mætning

KEYWORDS: Automated Theorem Proving - Intuitionistic Logic

FUNCTIONAL DESCRIPTION

Mætning is an automated theorem prover for first-order intuitionistic logic that is particularly suited for efficiently finding *disproofs*, i.e., for establishing that a given goal query is not provable. It is based on the focused inverse method [54] [74], but augmented by a mechanism for building finite approximations for infinite search spaces that nevertheless guarantee soundness of a disproof.

Mætning has been released under the terms of the MIT license.

- Participants: Taus Brock-Nannestad, Kaustuv Chaudhuri
- Contact: Kaustuv Chaudhuri
- URL: https://github.com/chaudhuri/maetning

6.5. Psyche

KEYWORDS: Proof-search - Correct-by-construction approach - Programmable Theorem Proving
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 small kernel interacts with plugins and decision procedures. The major effort in 2015 was a complete redesign of its architecture to allow the safe cooperation of various procedures (for e.g. different theories).

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

PI.R2 Project-Team

5. New Software and Platforms

5.1. Coq

KEYWORDS: Proof - Certification - Formalisation

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, ...

- Participants: Benjamin Grégoire, Enrico Tassi, Bruno Barras, Yves Bertot, Pierre Boutillier, Xavier Clerc, Pierre Courtieu, Maxime Dénès, Stéphane Glondu, Vincent Gross, Hugo Herbelin, Pierre Letouzey, Assia Mahboubi, Julien Narboux, Jean-Marc Notin, Christine Paulin-Mohring, Pierre-Marie Pédrot, Loïc Pottier, Matthias Puech, Yann Régis-Gianas, François Ripault, Matthieu Sozeau, Arnaud Spiwack, Pierre-Yves Strub, Benjamin Werner, Guillaume Melquiond and Jean-Christophe Filliâtre
- Partners: CNRS - Université Paris-Sud - ENS Lyon - Université Paris-Diderot
- Contact: Hugo Herbelin
- URL: http://coq.inria.fr/

5.1.1. Version 8.5

Cf. Highlights section. Version 8.5 includes as well a number of miscellaneous changes, at the level of tactics, of the specification language, of the Coq tools, of the standard library, altogether amounting to about 150 items in the change log of the version. In particular, Pierre-Marie Pédrot has been working on the overall optimisation of Coq, by tracking hotspots in the code. Coq v8.5 is currently much more efficient than its v8.4 counterpart, and is about as quick as v8.3, while having been expanded with a lot of additional features.

As a counterpart, the complexity of this new version induced a long phase of experimentation which included 3 different beta versions spanned over the whole 2015 year, with the final version being eventually released for the CoqPL workshop in January 2016.

5.1.2. Universes

Matthieu Sozeau followed up his work on universe polymorphism and uncovered important theoretical and practical problems regarding conversion and unification of universe polymorphic definitions in the presence of cumulativity and the Prop $\leq$ Type rule, as well as the invariants of the consistency checker. He also collaborated with Maxime Dénès and Benjamin Grégoire (Gallium and Marelle) on adapting the efficient conversion tests to universe polymorphism and with Enrico Tassi (Marelle) on the integration with the asynchronous proof development infrastructure. The universe polymorphic system is part of the 8.5 release.

5.1.3. The Equations plugin

Matthieu Sozeau continued work on the Equations plugin and fixed the remaining bugs preventing full automation of a middle-size example of formalisation – the normalisation proof of a predicate version of System F – together with Cyprien Mangin, during his master's internship. This involved finding a new termination proof for the calculus and making the dependent pattern-matching compilation more robust and axiom-free, using a different encoding of pattern-matching problems. This work was presented at LFMTP'15 in Berlin [29]. Since then, the system has been adapted to work with universe polymorphism and the new features of Coq 8.5.

5.1.4. Proof development in Coq

Pierre Letouzey developed a few new results about some Hofstadter sequences (see https://oeis.org/A005206 and https://oeis.org/A123070). These results have been proved in Coq, and they are presented in the technical report [39].

5.1.5. Proofs of algorithms on graphs

Chen Ran (ISCAS/SKLCS, Beijing) and Jean-Jacques Lévy pursued their work about producing readable formal proofs of graph algorithms. This work is performed in Why3 and partly in Coq. Graph algorithms are a good testbed for experimenting correctness proofs of programs with shared structures. We considered basic algorithms such as depth-first-search, random walk, acyclicity test, articulation points, strongly connected components, minimum spanning trees. In each case, the goal is to provide a simple proof as abstract as possible, although checked by computer. A longer term objective is to give formal proofs which could be inserted in algorithms textbooks. A progress work paper is under submission [41].

5.1.6. Development of programs for parallel and cloud computing

Frédéric Loulergue continued his work on the SyDPaCC framework. The goal of this framework is to ease the systematic development of correct parallel programs, in particular large-scale data-intensive applications. The parallel versions of the programs are written with a Coq axiomatisation of Bulk Synchronous Parallel ML (BSML) primitives. New results about SyDPaCC include the design and implementation of a new version of the core of the framework [21]. This new version has been used in a course of École des Jeunes Chercheur/se/s en Informatique Mathématique (EJCIM 2015) [38].

As the SyDPaCC framework currently mixes certified code extracted from Coq and unverified code, Frédéric Loulergue and Pierre Letouzey have worked on an extended extraction that generates, when possible, OCaml conditions for preconditions on function arguments. This part is still on-going work.

Frédéric Loulergue collaborated with Frédéric Dabrowski and Thomas Pinsard (Univ. Orléans) on the semantics and compilation of languages with nested atomic sections and thread escape. In [18], the focus is on the semantics of programming languages providing these features. The main contribution is the precise definition of atomicity, well-synchronisation and the proof that the latter implies the strong form of the former. A formalisation of the results in the Coq proof assistant is described.

In [27], the compilation of a language with nested atomic sections and thread escape towards a language with threads and locks is addressed. The design decisions of this compilation pass and of the target language were made with respect to the ultimate goal of a mechanised proof of semantic preservation.

Frédéric Loulergue collaborated with Allan Blanchard, Nikolai Kosmatov and Matthieu Lemerre (CEA LIST) on the verification of a critical component of a hypervisor. In [23], they present a case study on formal verification of the virtual memory system of the cloud hypervisor Anaxagoros, a microkernel designed for resource isolation and protection. The code under verification is specified and proven in the software verification framework, mostly using automatic theorem proving. The remaining properties are interactively proven with the Coq proof assistant.

Frédéric Loulergue collaborated with Asma Guesmi, Pascal Berthomé and Patrice Clemente (INSA Centre Val de Loire) on resources placement in the Cloud taking into account security requirements [28].

5.2. Other software developments

In collaboration with François Pottier (Inria Gallium), Yann Régis-Gianas maintained Menhir, an LR parser generator for OCaml. Yann Régis-Gianas develops the “Hacking Dojo”, a web platform to automatically grade programming exercises. The platform is now used in several courses of the University Paris Diderot. He gets help from the internship of Alexandre Ly, a master student of the Paris Diderot University. In collaboration with Beta Ziliani (LIIS, Cordoba, Argentine), Yann Régis-Gianas, Béatrice Carré and Jacques-Pascal Deplaix develop MetaCoq, an extension of Coq to use Coq as a metalanguage for itself.

SUMO Project-Team

6. New Software and Platforms

6.1. SIMSTORS

SIMSTORS is a simulator for regulated stochastic timed Petri nets. These Petri nets are a variant of stochastic and timed nets, which execution is controlled by a regulation policy an a predetermined theoretical schedule. The role of the regulation policy is to control the system to realize the schedule with the best possible precision. This software allows not only for step by step simulation, but also for performance analysis of systems such as production cells or train systems.

SIMSTORS was used successfully during a collaboration with Alstom transport to model existing urban railway systems and their regulation schemes. Alstom transport is willing to transfer this software and use it during early design phase of regulation algorithms in their metro lines.

Future extensions of the software will deal with verification of several new properties such as the robustness of proposed schedules.

- Participants: Loïc Hélouët and Abd El Karim Kecir
- Contact: Loïc Hélouët

6.2. Sigali

FUNCTIONAL DESCRIPTION

Sigali is a model-checker 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

6.3. Tipex

TImed Properties Enforcement during eXecution
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).

- Contact: Thierry Jéron, Hervé Marchand
- URL: http://srinivaspinisetty.github.io/Timed-Enforcement-Tools/

6.4. ReaX

ReaX is a tool developed by Nicolas Berthier that investigates the control of safety properties for infnite reactive synchronous systems modeled by arithmetic symbolic transition systems. It provides effective algorithms allowing to solve the safety control problem (including the dead-lock free case), and report some experiments. Its aim is to replace Sigali, which is limited to finite state systems described by boolean variables.

- Contact : Nicolas Berthier, Hervé Marchand
- URL : http://reatk.gforge.inria.fr/

6.5. Open Agora Core

Christophe Morvan participates to the implementation of a sophisticated voting system: Open Agora Core. It currently implements several voting methods among which *Condorcet* (Schulze method) or *instant runoff*. It is integrated into a Slack [0] polling plugin. This development serves as a basic building block in the process of elaborating Open Agora, a startup that should be created during 2016.

- Contact : Christophe Morvan
- URL : http://www.open-agora.com

[0] Slack, `http://slack.com`, is an industrial team communication tool.

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.

- Participants: Sylvain Conchon, Évelyne Contejean, Mohamed Iguernelala, Stéphane Lescuyer and Alain Mebsout
- 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 notation and tactics for manipulating characteristic formulae interactively in Coq.

- Contact: Arthur Charguéraud
- URL: http://www.chargueraud.org/softs/cfml/

6.3. Coq

KEYWORDS: Proof - Certification - Formalisation
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, ...

- Participants: Benjamin Grégoire, Enrico Tassi, Bruno Barras, Yves Bertot, Pierre Boutillier, Xavier Clerc, Pierre Courtieu, Maxime Denes, Stéphane Glondu, Vincent Gross, Hugo Herbelin, Pierre Letouzey, Assia Mahboubi, Julien Narboux, Jean-Marc Notin, Christine Paulin-Mohring, Pierre-Marie Pédrot, Loïc Pottier, Matthias Puech, Yann Régis-Gianas, François Ripault, Matthieu Sozeau, Arnaud Spiwack, Pierre-Yves Strub, Benjamin Werner, Guillaume Melquiond and Jean-Christophe Filliâtre
- Partners: CNRS - Université Paris-Sud - ENS Lyon - Université Paris-Diderot
- Contact: Hugo Herbelin
- 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. CoqInterval provides a method for proving automatically the inequality of two expression of real values.

The Interval package provides several tactics for helping a Coq user to prove 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 inside the Coq proof assistant, with a special focus on genericity and efficiency for the computations. In 2014, this library has been included in CoqInterval.

- Participants: Guillaume Melquiond, Érik Martin-Dorel, Nicolas Brisebarre, Miora Maria Joldes, Micaela Mayero, Jean-Michel Muller, Laurence Rideau and Laurent Théry
- Contact: Guillaume Melquiond
- 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: Sylvie Boldo, Catherine Lelay and Guillaume Melquiond
- 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: Sylvain Conchon and Alain Mebsout
- 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 certify numerical applications.

Flocq is currently used by the CompCert certified compiler for its support of floating-point computations.

- Participants: Guillaume Melquiond and Sylvie Boldo
- Contact: Sylvie Boldo
- 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 verifying and formally proving properties on 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 certify 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.

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

VERIDIS Project-Team

6. New Software and Platforms

6.1. The Redlog Computer Logic System

FUNCTIONAL 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 has been publicly available since 1995 and is constantly being improved. The name Redlog stands for Reduce Logic System. Andreas Dolzmann from Schloss Dagstuhl Leibniz-Zentrum für Informatik is a co-developer of Redlog.

Reduce and Redlog are open-source and freely available under a modified BSD license at http://reduce-algebra.sourceforge.net/. The Redlog homepage is located at http://www.redlog.eu/. 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.

Effective quantifier elimination procedures for the various supported theories establish an important class of methods available in Redlog. For the theories supported by Redlog, quantifier elimination procedures immediately yield decision procedures. Besides these quantifier elimination-based decision methods there are specialized, and partly incomplete, decision methods, which are tailored to input from particular fields of application.

In 2015 there was further significant progress with the identification of bifurcations in biochemical models based on real reasoning [17], [33]. With existential real quantifier elimination Redlog can now produce unsatisfiable cores in the infeasible case [27]. This is of considerable relevance in the course of using Redlog as a theory solver in SMT contexts, e.g., within the SMArT project (section 9.2).

Redlog is a widely accepted tool and highly visible in mathematics, informatics, engineering and the sciences. The seminal article on Redlog [4] has received more than 300 citations in the scientific literature so far.

- Participants: Thomas Sturm, Marek Kosta, and Maximilian Jaroschek
- Contact: Thomas Sturm
- URL: http://www.redlog.eu/

6.2. SPASS

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. It has been developed since the mid-1990s at the Max-Planck Institut für Informatik in Saarbrücken. Version 3.8 is the final release of the SPASS first-order prover built on a traditional "select given loop" design; it is distributed under the FreeBSD license at http://www.spass-prover.org.

SPASS will be released in the future in the form of various reasoners for different logics, including combinations of first-order logic with background theories, in particular some forms of arithmetic. In 2015, we have continued our efforts to improve the superposition calculus as well as to develop dedicated arithmetic decision procedures for various arithmetic theories, in particular linear integer arithmetic. Our results are:

- new calculi and decidability results for finite domain fragments,
- specialized reasoning support for finite subsets,
- specialized decision procedures for linear real arithmetic with one quantifier alternation,
- new efficient and complete procedures for (mixed) linear integer arithmetic,
- decidability results and respective procedures for various combinations of linear arithmetic with first-order logic.
- Participants: Martin Bromberger, Thomas Sturm, Marco Voigt, Uwe Waldmann, Christoph Weidenbach
- Contact: Christoph Weidenbach
- URL: http://www.spass-prover.org/

6.3. The TLA+ Proof System

FUNCTIONAL DESCRIPTION

TLAPS, the TLA$^+$ proof system developed at the Joint MSR-Inria Centre, 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 independent proof steps. 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$^+$ 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.

The current version 1.4.3 of TLAPS was released in June 2015, it is distributed under a BSD-like license. The prover fully handles the non-temporal part of TLA$^+$. Basic temporal logic reasoning is supported through an interface with a decision procedure for propositional temporal logic that performs on-the-fly abstraction of first-order subformulas. Symmetrically, subformulas whose main operator is a connective of temporal logic are abstracted before being sent to backends for first-order logic.

A complete rewrite of the proof manager has started in 2015. Its objectives are to replace the ad-hoc parser used so far with an interface to SANY, the standard parser and semantic analyzer for TLA$^+$, to extend the scope of the fragment of TLA$^+$ that is handled by TLAPS, and general code refactoring and performance improvements.

TLAPS has been used in several case studies in 2015, including the proof of determinacy of PharOS (section 8.1) and the verification of the Pastry routing protocol (section 7.2). These case studies feed back into the standard library of the distribution.

- Participants: Stephan Merz, Martin Riener, Hernán Vanzetto
- Contact: Stephan Merz
- URL: http://tla.msr-inria.inria.fr/tlaps/content/Home.html

6.4. The veriT Solver

FUNCTIONAL DESCRIPTION

VeriT is an open, trustable and efficient SMT (Satisfiability Modulo Theories) solver developed in cooperation with David Déharbe from the Federal University of Rio Grande do Norte in Natal, Brazil, on leave for Clearsy. The solver can handle large quantifier-free formulas containing uninterpreted predicates and functions, and arithmetic over integers and reals. It features efficient decision procedures for uninterpreted symbols and linear arithmetic. It also has some support for user-defined theories, quantifiers, and lambda-expressions. This allows users to easily express properties about concepts involving sets, relations, etc. The prover can produce explicit proof traces when it is used as a decision procedure for quantifier-free formulas with uninterpreted symbols and arithmetic. To support the development of the tool, non-regression tests use Inria's grid infrastructure; it allows us to extensively test the solver on thousands of benchmarks in a few minutes. The veriT solver is available as open source under the BSD license at the veriT Web site.

Efforts in 2015 have been focused on efficiency, stability, and expressiveness, with a new ability for handling non-linear arithmetic. The decision procedures for uninterpreted symbols and linear arithmetic have been further improved. The integration of the solver Redlog (section 6.1) for non-linear arithmetic in the context of the SMArT project (section 9.2) now works for quantifier-free formulas with non-linear real arithmetic, but is not yet complete for combinations.

The veriT solver participated in the SMT competition SMT-COMP 2015 with decent 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 for discharging proof obligations generated in Event-B [53]; on a large repository of industrial and academic cases, this SMT-based plugin decreased by 75% the number of proof obligations requiring human interactions, compared to the original B prover.

- Participants: Pascal Fontaine, Pablo Dobal, David Déharbe, and Haniel Barbosa
- Partners: Université de Lorraine - Federal University of Rio Grande do Norte
- Contact: Pascal Fontaine
- URL: http://www.veriT-solver.org

CARTE Project-Team

6. New Software and Platforms

6.1. CoDisasm

FUNCTIONAL DESCRIPTION

Codisasm is a new disassembly program which supports self-modifying code and code overlapping. Up to our knowledge, it is the first which copes both aspects of program obfuscation. The tool is based on the notion of "wave" developed in the group.
It is written in C and contains about 3k lines of code.

- Contact: Fabrice Sabatier
- URL: http://www.lhs.loria.fr/wp/?p=289

6.2. DynamicTracer

FUNCTIONAL DESCRIPTION

DynamicTracer is a new tool with a public web interface which provides run traces of executable files. The trace is obtained by recording a dynamic execution in a safe environment. It contains instruction addresses, instruction opcodes and other optional information.
It is written in C++ and contains about 2.5k lines of code.

- Contact: Fabrice Sabatier
- URL: http://www.lhs.loria.fr

6.3. Gorille

FUNCTIONAL DESCRIPTION

Gorille (formerly MMDEX) is a virus detector based on morphological analysis. It is composed of our own disassembler tool, of a graph transformer and a specific tree-automaton implementation. The tool is used in the EU-Fiware project and by some other partners (e.g., DAVFI project).
It is written in C and contains about 100k lines of code.
APP License, IDDN.FR.001.300033.000.R.P.2009.000.10000, 2009.

- Contact: Philippe Antoine
- URL: http://www.lhs.loria.fr

CASSIS Project-Team

6. New Software and Platforms

6.1. Protocol Verification Tools

Participants: Véronique Cortier, Stéphane Glondu, Pierre-Cyrille Héam, Olga Kouchnarenko, Steve Kremer, Michaël Rusinowitch, Mathieu Turuani, Laurent Vigneron.

6.1.1. CL-AtSe

We develop *CL-AtSe*, a Constraint Logic based Attack Searcher for cryptographic protocols, initiated and continued by the European projects *AVISPA*, AVANTSSAR (for web-services) and Nessos respectively. The *CL-AtSe* approach to verification consists in a symbolic state exploration of the protocol execution for a bounded number of sessions, thus is both correct and complete. *CL-AtSe* includes a proper handling of sets, lists, choice points, specification of any attack states through a language for expressing e.g., secrecy, authentication, fairness, or non-abuse freeness, advanced protocol simplifications and optimizations to reduce the problem complexity, and protocol analysis modulo the algebraic properties of cryptographic operators such as XOR (exclusive or) and Exp (modular exponentiation).

CL-AtSe has been successfully used to analyse protocols from e.g., France Telecom R&D, Siemens AG, IETF, Gemalto, Electrum in funded projects. It is also employed by external users, e.g., from the AVISPA's community. Moreover, *CL-AtSe* achieves good analysis times, comparable and sometimes better than other state-of-the art tools.

CL-AtSe has been enhanced in various ways. It fully supports the Aslan semantics designed in the context of the AVANTSSAR project, including Horn clauses (for intruder-independent deductions, e.g., for credential management), and a large fragment of LTL-based security properties. A Bugzilla server collects bug reports, and online analysis and orchestration are available on our team server (https://cassis.loria.fr). Large models can be analysed on the TALC Cluster in Nancy with parallel processing. *CL-AtSe* also supports negative constraints on the intruder's knowledge, which reduces drastically the orchestrator's processing times and allows separation of duties and non-disclosure policies, as well as conditional security properties, like: i) an authentication to be verified iff some session key is safe; ii) relying on a leaking condition on some private data instead of an honesty predicate to trigger or block some agent's property. This was crucial for e.g., the Electrum's wallet where all clients can be dishonest but security guarantees must be preserved anyway.

6.1.2. Akiss

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. The tool also include the possibility for checking everlasting indistinguishability properties [63].

The tool is still under active development, including optimisations to improve efficiency, but also the addition of new features, such as the possibility to model protocols using weak secrets, and the addition of support for exclusive or.

The *Akiss* tool is freely available at https://github.com/akiss/akiss.

6.1.3. Belenios

In collaboration with the Caramel project-team, we develop an open-source private and verifiable electronic voting protocol, named *Belenios*. Our system is an evolution and a new implementation of an existing system, Helios, developed by Ben Adida, and used e.g., by UCL and the IACR association in real elections. The main differences with Helios are a cryptographic protection against ballot stuffing and a practical threshold decryption system that allows to split the decryption key among several authorities, k out of n authorities being sufficient to decrypt. We will continue to add new cryptographic and protocol improvements to offer a secure, proved, and practical electronic voting system.

Belenios has been implemented (cf. http://belenios.gforge.inria.fr) by Stéphane Glondu and has been tested in December 2014 "in real conditions", in a test election involving the members of Inria Nancy-Grand Est center and of the Loria lab (more than 500 potential voters) that had to elect the best pictures of the Loria. Since 2015, it is used by the CNRS for remote election among its councils. It has also been used to elect the leader of the C2 GdR-IM working group [0] (about 230 voters and 100 ballots cast). It has also been used in some smaller elections (eg to chose an invited speaker).

6.1.4. SAPIC

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 be analysed using the Tamarin Prover.

Its aim is 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 has been succesfully applied on several case studies including the Yubikey authentication protocol.

A recent extension, SAPIC* extends *SAPIC* by a Kleene star operator (*) which allows to iterate a process a finite but arbitrary number of times. This construction is useful to specify for instance stream authentication protocols. We used it to analyse a simple version of the TESLA protocol.

SAPIC is freely available at http://sapic.gforge.inria.fr/.

6.2. Testing Tools

Participants: Fabrice Bouquet, Frédéric Dadeau, Elizabeta Fourneret.

6.2.1. Hydra

Hydra is an Eclipse-like platform, based on Plug-ins architecture. Plug-ins can be of five kinds: *parser* is used to analyze source files and build an intermediate format representation of the source; *translator* is used to translate from a format to another or to a specific file; *service* denotes the application itself, i.e., the interface with the user; *library* denotes an internal service that can be used by a service, or by other libraries; *tool* encapsulates an external tool. The following services have been developed so far:

- BZPAnimator: performs the animation of a BZP model (a B-like intermediate format);
- Angluin: makes it possible to perform a machine learning algorithm (à la Angluin) in order to extract an abstraction of a system behavior;
- UML2SMT: aims at extracting first order logic formulas from the UML Diagrams and OCL code of a UML/OCL model to check them with a SMT solver.

These services involve various libraries (sometimes reusing each other), and rely on several *tool* plug-ins that are: SMTProver (encapsulating the Z3 solver), PrologTools (encapsulating the CLPS-B solver), Grappa (encapsulating a graph library). We are currently working on transferring the existing work on test generation from B abstract machines, JML, and statecharts using constraint solving techniques.

[0] https://crypto.di.ens.fr/c2:election

6.2.2. jMuHLPSL

jMuHLPSL [6] is a mutant generator tool that takes as input a verified HLPSL protocol, and computes mutants of this protocol by applying systematic mutation operators on its contents. The mutated protocol then has to be analyzed by a dedicated protocol analysis tool (here, the AVISPA tool-set). Three verdicts may then arise. The protocol can still be *safe*, after the mutation, this means that the protocol is not sensitive to the realistic "fault" represented by the considered mutation. This information can be used to inform the protocol designers of the robustness of the protocol w.r.t. potential implementation choices, etc. The protocol can also become *incoherent*, meaning that the mutation introduced a functional failure that prevents the protocol from being executed entirely (one of the participants remains blocked in a given non-final state). The protocol can finally become *unsafe* when the mutation introduces a security flaw that can be exploited by an attacker. In this case, the AVISPA tool-set is able to compute an attack-trace, that represents a test case for the implementation of the protocol. If the attack can be replayed entirely, then the protocol is not safe. If the attack can not be replayed then the implementation does not contain the error introduced in the original protocol.

The tool is written in Java, and it is freely available at: http://members.femto-st.fr/sites/femto-st.fr.frederic-dadeau/files/content/pub/jMuHLPSL.jar.

6.2.3. Praspel

Praspel is both a specification language, a test data generator and test execution driver for PHP programs. These latter are annotated to describe class (resp. method) contracts using invariants (resp. pre- and postconditions). Praspel contracts allow to describe data typing informations, by means of *realistic domains*. According to the contract-driven testing principles, the tool uses the contracts to both generate test data, using dedicated test generators (random for integer variables, grammar-based for strings, constraint-based for arrays), and establish the test verdict by checking the contract assertions at run-time.

The tool is open source and freely available at: http://hoa-project.net. It has been integrated into a PHP framework named Hoa, and coupled with the atoum tool (https://github.com/atoum/atoum) that can be used to execute the tests and report on their code coverage.

6.3. Other Tools

Several software tools described in previous sections are using tools that we have developed in the past. For instance BZ-TT uses the set constraints solver CLPS. Note that the development of the SMT prover haRVey has been stopped. The successor of haRVey is called veriT and is developed by David Déharbe (UFRN Natal, Brasil) and Pascal Fontaine (Veridis team). We have also developed, as a second back-end of *AVISPA*, TA4SP (Tree Automata based on Automatic Approximations for the Analysis of Security Protocols), an automata based tool dedicated to the validation of security protocols for an unbounded number of sessions.

We have also designed tools to manage collaborative works on shared documents using flexible access control models. These tools have been developed in order to validate and evaluate our approach on combining collaborative edition with optimistic access control.

COMETE Project-Team

6. New Software and Platforms

6.1. Location Guard

Participants: Konstantinos Chatzikokolakis [correspondant], Marco Stronati.

https://github.com/chatziko/location-guard

The purpose of Location Guard is to protect the user's location during the use of a location-based service, in an easy and intuitive way that makes it 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.

Although both mobile operating systems and browsers require the user's permission to disclose location information, the user faces an "all-or-nothing" choice: either disclose his exact location and give up his privacy, or stop using the application. This forces many users to disclose their location, although ideally they would like to enjoy some privacy.

The API level of a browser or an operating system is an ideal place for integrating a location obfuscation technique, in a way that is easy to understand for the average user, and readily available to all applications. When an application asks for the user's location, the browser or operating system can ask the user's permission, but including the option to provide an obfuscated location instead of the real one! Different levels of obfuscation can be also offered, so that the user can chose to provide more accurate location to applications that really need it, and more noisy location to those that don't.

In 2015, Location Guard matured with several additions and fixes throughout the year, and was selected by Mozilla as the pick of the month for June 2015, confirming the users' general interest in location privacy.

Moreover in 2015 we set the foundations for actively using Location Guard as a platform for performing research on location privacy. Since location data are sensitive, since the creation of Location Guard we chose to collect no data whatsoever from the users. However, such data are invaluable for research purposes. As a consequence, we created a framework for *locally* collecting data at the user's machine, perform an analysis also locally, and collect back only the results of the analysis for research purposes.

DECENTRALISE Team

5. New Software and Platforms

5.1. GNUnet

GNUnet
KEYWORD: Privacy
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: Hans Grothoff, Florian Dold, Jeffrey Paul Burdges and Gabor Toth
- Partner: The GNU Project
- Contact: Hans Grothoff
- URL: https://gnunet.org/

5.2. MHD

GNU libmicrohttpd
KEYWORDS: Embedded - Web 2.0
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.

- Author: Hans Grothoff
- Contact: Hans Grothoff
- URL: http://www.gnu.org/software/libmicrohttpd/

5.3. Taler

GNU Taler
KEYWORD: Privacy
FUNCTIONAL DESCRIPTION

Taler is a new electronic payment system.

- Partner: The GNU Project
- Contact: Hans Grothoff
- URL: http://taler.net/

DICE Team

5. New Software and Platforms

5.1. BitBallot

The BitBallot voting protocol is designed to avoid the concentration of data by third party. The protocol allows users to cast their ballot on their mobile device, and then share only restricted amounts of their data with other peers to compute the tally. Unlike other protocols, voters pull data from others instead of pushing their own votes.

Convinced by the need of new election mechanisms, to support emerging forms of more continuous democracy, we are developing BitBallot, to allow elections with distributed tallying that incorporate individual verification. As such, it provides anonymity of the data sources, non interruptible run-time, global access to results, and non-predictability of results through partial communication spying. Cryptography is not essential to protect the privacy of the voters or the secrecy of the ballots. On the basis of this protocol, a SaaS platform that allows to run public tests online is under development.

- Contact: Stéphane Grumbach, Stéphane Frénot, Damien Reimert, Robert Riemann

5.2. C3PO

Social networks put together individuals with common interests and/or existing real-life relationships so that they can produce and share information. There is a strong interest of individuals towards these networks. They rely in general on a stable, centralized network infrastructure, and a user will always be provided with the same services no matter what their current context is. By contrast, the C3PO project (C3PO stands for Collaborative Creation of Contents and Publishing using Opportunistic networks) aims at promoting “spontaneous and ephemeral social networks” (SESN), built on top of a peer-to-peer distributed architecture leveraging ad-hoc mobile networks and the resources and services offered by mobile devices. As with traditional social networks, SESN can put together nomad individuals based on their affinities and common interests so that they can collaboratively work on tasks as part of a SESN. (Supported by an ANR project.)

- Contact: Stéphane Frénot, Damien Reimert

5.3. Fluxion

This joint project with Worldline aims at managing mobile code in complex Web architectures. We design a fast and reactive framework, transparently moving functions between running systems to cope with the load variation in high performance Web architectures. The Fluxion model is our approach to design mobile application modules that are a mix of functional programming and flow based reactive systems. We work on compilation techniques to transform a Javascript event-loop into a parallelized pipeline where each stage is made independent from the main event-loop.

- Contact: Stéphane Frénot, Etienne Brodu

5.4. Jumplyn

Jumplyn is a student project delivery platform. It offers a service based on three features: the ongoing management of the project, resources recommendation, and enhancement of the activity. Like any intermediation platform, it speaks directly to its users, students, and puts them in relation to relevant information.

- Contact: Stéphane Frénot, Stéphane Grumbach, Auguste Caen
- URL: http://www.jumplyn.com

PRIVATICS Project-Team

5. New Software and Platforms

5.1. Mobilitics

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, Vincent Roca, Gwendal Le Grand, Geoffrey Delcroix, Franck Baudot and Stéphane Petitcolas
- Contact: Claude Castelluccia
- URL: https://team.inria.fr/privatics/fr/mobilitics/

5.2. 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: Pierre Rouveyrol and Claude Castelluccia
- Contact: Claude Castelluccia

5.3. OPENFEC

FUNCTIONAL DESCRIPTION

OpenFEC is an open-source C-language implementation of several Application-Level Forward Erasure Correction (AL-FEC) codecs, namely: 2D-parity, Reed-Solomon (RFC 5510) and LDPC-Staircase (RFC 5170) codes. The OpenFEC project also provides a complete performance evaluation tool-set, capable of automatically assessing the performance of various codecs, both in terms of erasure recovery and encoding/decoding speed or memory consumption.

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

5.4. FECFRAME

FUNCTIONAL DESCRIPTION

FECFRAME implements IETF FECFRAME (RFC 6363). It allows to transmit multimedia streams to one or severals receivers at the same time while being robust to packet losses occurring on the network (par ex. 3G/4G or Wifi). This software is compatible with OpenFec which provides error-correcting codes.

- Participants: Vincent Roca
- Contact: Vincent Roca

5.5. WALTER

Walter experiment: "Is My Web Content Altered?". A web based tool detecting the unwanted injection of scripts and other contents in unencrypted webpages.

FUNCTIONAL DESCRIPTION

Disputable network agents, namely free Wi-Fi hotspots providers such as those found in airports or coffee shops, have been found to monetize their networks by injecting advertisements and trackers into their customers' traffic. Such adverts are served by network agents instead of website publishers. This is a relatively new approach, and we are trying to determine its usage worldwide. This website is designed to assess whether your internet connection is affected by such practices. We also detect local page alterations that come from browser extensions and programs that may run on your machine.

- Participants: Mathieu Cunche, Leo Letaro.
- Contact: Mathieu Cunche

PROSECCO Project-Team

6. New Software and Platforms

6.1. ProVerif

Participants: Bruno Blanchet [correspondant], Xavier Allamigeon [April–July 2004], Vincent Cheval [Sept. 2011–], Benjamin Smyth [Sept. 2009–Feb. 2010].

PROVERIF (http://proverif.inria.fr) 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 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.

The PROVERIF verifier can prove the following properties:

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

PROVERIF is widely used by the research community on the verification of security protocols (see http://proverif.inria.fr/proverif-users.html for references).

PROVERIF is freely available on the web, at http://proverif.inria.fr, under the GPL license.

6.2. CryptoVerif

Participants: Bruno Blanchet [correspondant], David Cadé [Sept. 2009–].

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

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

CRYPTOVERIF has been used in particular for a study of Kerberos in the computational model, and as a back-end for verifying implementations of protocols in F# and C.

CRYPTOVERIF is freely available on the web, at http://cryptoverif.inria.fr, under the CeCILL license.

6.3. miTLS

Participants: Karthikeyan Bhargavan [correspondant], Antoine Delignat-Lavaud, Cedric Fournet [Microsoft Research], Markulf Kohlweiss [Microsoft Research], Alfredo Pironti, Pierre-Yves Strub [IMDEA], Santiago Zanella-Béguelin [Microsoft Research], Jean Karim Zinzindohoue.

miTLS is a verified reference implementation of the TLS security protocol in F#, a dialect of OCaml for the .NET platform. It supports SSL version 3.0 and TLS versions 1.0-1.2 and interoperates with mainstream web browsers and servers. miTLS has been verified for functional correctness and cryptographic security using the refinement typechecker F7.

A paper describing the miTLS library was published at IEEE S&P 2013, CRYPTO 2014, and several updates to the software were released in 2015. The software and associated research materials are available from http://mitls.org.

6.4. flexTLS

Participants: Karthikeyan Bhargavan [correspondant], Alfredo Pironti, Benjamin Beurdouche.

flexTLS is a TLS testing framework based on miTLS, and is released as part of the miTLS distribution. Unlike miTLS, flexTLS can be configured to run incorrect TLS clients and servers in order to test other TLS implementations. Using flexTLS we analyzed a series of open source TLS implementations and found important vulnerabilities like SKIP and FREAK. We also used flexTLS to build proof-of-concept demos for other attacks such as Logjam.

A paper describing flexTLS was published at Usenix WOOT 2015. The software and associated research materials are available from http://mitls.org.

6.5. F*

Participants: Nikhil Swamy [Microsoft Research], Karthikeyan Bhargavan, Antoine Delignat-Lavaud, Cedric Fournet [Microsoft Research], Catalin Hritcu, Chantal Keller, Aseem Rastogi, Pierre-Yves Strub.

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 Fω (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.

A detailed description of F* (circa 2011) appeared in the Journal of Functional Programming [53]. F* has evolved substantially since then. The latest version of F* is written entirely in F*, and bootstraps in OCaml and F#. It is under active development at GitHub: https://github.com/FStarLang and the official webpage is at http://fstar-lang.org.

6.6. ProScript

Participants: Nadim Kobeissi [correspondant], Karthikeyan Bhargavan, Bruno Blanchet.

Defensive JavaScript (DJS) is a subset of the JavaScript language that guarantees the behaviour of trusted scripts when loaded in an untrusted web page. Code in this subset runs independently of the rest of the JavaScript environment. When propertly wrapped, DJS code can run safely on untrusted pages and keep secrets such as decryption keys. ProScript is a typed subset of JavaScript, inspired by DJS, that is focused on writing verifiable cryptographic protocol implementations. In addition to DJS typing, ProScript imposes a functional style that results in more readable and easily verifiable ProVerif models. ProScript has been used to write and verify a full implementation of the TextSecure protocol in JavaScript.

The ProScript compiler and various libraries written in ProScript will be made available from the Prosecco webpage.