RESEARCH CENTER
Paris - Rocquencourt

FIELD

Activity Report 2013

Section Software

Edition: 2014-03-19

AXIS Project-Team

5. Software and Platforms

5.1. Introduction

From its creation, AxIS has proposed new methods and software validated experimentally on various applications: Data Mining, Web Usage Mining, Information Retrieval, Activity Modeling. See Sections from 5.3 until 5.6 and our 2013 results.

In the context of the CPER Télius contract (2010-2013), AxIS has proposed to provide a Focus platform (renamed FocusLab) aiming the community of Living Labs or any researcher/actor involving in experimental project with users.

5.2. FocusLab Platform

Participant: Brigitte Trousse [co-correspondent].

Between 2010-2012 in the context of CPER Télius (cf. Section 7.1.1), we bought various hardware (eye-trackers, physiologic sensors, equilibrium platform, tablets, Arduino components, etc.) and software (Sphinx for questionnaires, Story Board for usage scenarios, Interface prototyping tools such as JustInMind, etc.) in order to observe and analyse user behaviours in supporting the design and evaluation of ICT-based services or products within a living lab approach and also to mine data. FocusLab hardware and software (under licences) were used since 2011 by Inria teams and external collaborators with success, supporting experiments or training.
Our goal was also to provide a web-based application for reserving FocusLab material (hardware, software and documentation) and to prepare the access/download of some software issued from Inria research. We started with AxIS software as a first step of the mutualised software part of the platform.
The development process of the web-based FocusLab platform started slowly in 2011, after finding some ways to fund human resources. We started by transforming some AxIS KDD methods into web services. Such a work was pursued this year (cf. Section 6.6) linked to Elliot purposes. This platform (http://focuslab.inria.fr) is based on a Service oriented Architecture.

5.3. Data Mining

5.3.1. Classification and Clustering Methods

Participants: Marc Csernel, Yves Lechevallier [co-correspondent], Brigitte Trousse [co-correspondent].

We developed and maintained a collection of clustering and classification software, written in C++ and /or Java:

Supervised methods

- a Java library (Somlib) that provides efficient implementations of several SOM(Self-Organizing Map) variants [44], [43], [69], [68], [73], especially those that can handle dissimilarity data (available on Inria's Gforge server (public access) Somlib, developed by AxIS Rocquencourt and Brieuc Conan-Guez from Université de Metz.
- a functional Multi-Layer Perceptron library, called FNET, that implements in C++ supervised classification of functional data [64], [67], [66], [65] (developed by AxIS Rocquencourt).

Unsupervised methods : partitioning methods

- Two partitioning clustering methods on the dissimilarity tables issued from a collaboration between AxIS Rocquencourt team and Recife University, Brazil: CDis and CCClust [77]. Both are written in C++ and use the "Symbolic Object Language" (SOL) developed for SODAS. And one partitioning method on interval data (Div).
- Two standalone versions improved from SODAS modules, SCluster and DIVCLUS-T [41] (AxIS Rocquencourt).

Unsupervised methods : agglomerative methods

- a Java implementation of the 2-3 AHC (developed by AxIS Sophia Antipolis). The software is available as a Java applet which runs the hierarchies visualization toolbox called HCT for Hierarchical Clustering Toolbox (see [3] and [42]).

A Web interface developed in C++ and running on our Apache internal Web server .is available for the following methods: SCluster, Div, Cdis, CCClust.
Previous versions of the above software have been integrated in the SODAS 2 Software [61] which was the result of the european project ASSO [6] (2001-2004). SODAS 2 supports the analysis of multidimensional complex data (numerical and non numerical) coming from databases mainly in statistical offices and administration using Symbolic Data Analysis [39]. This software is registrated at APP (Agence de la Protection des Programmes). For the latest version of the SODAS 2 software, see [60], [79].

In 2013, a new release of MND (Dynamic Clustering Method for Mixed Data) algorithm has been done based on [80] (cf. section 6.2.5) and used on clustering the user profiles and analysing user behaviour change (cf. Section 6.5.4).

5.3.2. *Extracting Sequential Patterns with Low Support*

Participant: Brigitte Trousse [correspondent].

Two methods for extracting sequential patterns with low support have been developed by D. Tanasa in his thesis (see Chapter 3 in [72] for more details) in collaboration with F. Masseglia and B. Trousse :

- **Cluster & Divide**,
- and **Divide & Discover** [8].

These methods have been successfully applied from 2005 on various Web logs.

5.3.3. *Mining Data Streams*

Participant: Brigitte Trousse [correspondent].

In Marascu's thesis (2009) [57], a collection of software have been developed for knowledge discovery and security in data streams. Three **clustering methods for mining sequential patterns (Java) in data streams** method have been developed in Java:

- SMDS compares the sequences to each others with a complexity of _$O(n^2)$.
- SCDS is an improvement of SMDS, where the complexity is enhanced from $O(n^2)$ to $O(n.m)$ with n the number of navigations and m the number of clusters.
- ICDS is a modification of SCDS. The principle is to keep the clusters' centroids from one batch to another.

Such methods take batches of data in the format "Client-Date-Item" and provide clusters of sequences and their centroids in the form of an approximate sequential pattern calculated with an alignment technique.

In 2010 the Java code of one method called SCDS has been integrated in the MIDAS demonstrator and a C++ version has been implemented by F. Masseglia for the CRE contract with Orange Labs with the deliverability of a licence) with a visualisation module (in Java).

[6] ASSO: Analysis System of Symbolic Official data.

It has been tested on the following data:

- Orange mobile portal logs (100 million records, 3 months) in the context of Midas project (Java version) and the CRE (Orange C++ version)
- Inria Sophia Antipolis Web logs (4 million records, 1 year, Java version)
- Vehicle trajectories (Brinkhoff generator) in the context of MIDAS project (Java version).

In 2012 within the context of the ELLIOT contract, SCDS has been integrated as a Web service (Java version) in the first version of FOCUSLAB platform: a demonstration was made on San Rafaelle Hospital media use case at the first ELLIOT review at Brussels. We applied SCDS web service on data issued from two other use cases in Logistics (BIBA) and Green Services (Inria) [38].

The three C++ codes done for the CRE (Orange Labs) have been deposit at APP. The java code will be deposit in 2014 at APP.

5.4. Web Usage Mining

5.4.1. AWLH for Pre-processing Web Logs

Participants: Yves Lechevallier [co-correspondent], Brigitte Trousse [co-correspondent].

AWLH (AxIS Web Log House) for Web Usage Mining (WUM) is issued from AxISLogMiner software which implements the mult-site log preprocessing methodology and extraction of sequential pattern with low support developed by D. Tanasa in his thesis [72] , [15] for Web Usage Mining (WUM). In the context of the Eiffel project (2008-2009), we isolated and redesigned the core of AxISlogMiner preprocessing tool (we called it AWLH) composed of a set of tools for pre-processing web log files. The web log files are cleaned before to be used by data mining methods, as they contain many noisy entries (for example, robots requests). The data are stored within a database whose model has been improved.

So AWLH offers:

- Processing of several log files from several servers,
- Support of several input formats (CLF, ECLF, IIS, custom, etc.),
- Incremental pre-processing,
- Java API to help integration of AWLH in external application.

5.4.2. ATWUEDA for Analysing Evolving Web Usage Data

Participants: Yves Lechevallier [co-correspondent], Brigitte Trousse [co-correspondent].

ATWUEDA for Web Usage Evolving Data Analysis [52] [4] was developed by A. Da Silva in her thesis [52] under the supervision of Y. Lechevallier. This tool was developed in Java and uses the JRI library in order to allow the application of R which is a programming language and software environment for statistical computing functions in the Java environment.

ATWUEDA is able to read data from a cross table in a MySQL database. It splits the data according to the user specifications (in logical or temporal windows) and then applies the approach proposed in the Da Silva's thesis in order to detect changes in dynamic environment. The proposed approach characterizes the changes undergone by the usage groups (e.g. appearance, disappearance, fusion and split) at each time-stamp. Graphics are generated for each analysed window, exhibiting statistics that characterizes changing points over time.

Version 2. of ATWUEDA (September 2009) is available at Inria's gforge website.

The efficiency of ATWUEDA [46] has been demonstrated by applying it on real case studies such as on condition monitoring data streams of an electric power plant provided by EDF.

ATWUEDA is used by Telecom Paris Tech and EDF [4].

5.5. Information Retrieval

5.5.1. CBR*Tools for Managing and Reusing Past Experiences based on Historical Data

Participant: Brigitte Trousse [correspondent].

CBR*Tools [53], [54] is an object-oriented framework [55], [50] for Case-Based Reasoning which is specified with the UMT notation (Rational Rose) and written in Java. It offers a set of abstract classes to model the main concepts necessary to develop applications integrating case-based reasoning techniques: case, case base, index, measurements of similarity, reasoning control. It also offers a set of concrete classes which implements many traditional methods (closest neighbours indexing, Kd-tree indexing, neuronal approach based indexing, standards similarities measurements). CBR*Tools currently contains more than 240 classes divided in two main categories: the core package for basic functionality and the time package for the specific management of the behavioural situations. The programming of a new application is done by specialization of existing classes, objects aggregation or by using the parameters of the existing classes.

CBR*Tools addresses application fields where the re-use of cases indexed by behavioural situations is required. The CBR*Tools framework was evaluated via the design and the implementation of several applications such as Broadway-Web, Educaid, BeCKB, Broadway-Predict, e-behaviour and Be-TRIP.

CBR*Tools is concerned by two past contracts: EPIA and MobiVIP.

CBR*Tools is available on demand for research, teaching and academic purpose via the FocusLab platform. The user manual can be downloaded at the URL: http://www-sop.inria.fr/axis/cbrtools/manual/.

See also the web page http://www-sop.inria.fr/axis/cbrtools/manual/.

5.5.2. Broadway*Tools for Building Recommender Systems on the Web

Participant: Brigitte Trousse [correspondent].

Broadway*Tools is a toolbox supporting the creation of adaptive recommendation systems on the Web or in a Internet/Intranet information system. The toolbox offers different servers, including a server that computes recommendations based on the observation of the user sessions and on the re-use of user groups' former sessions. A recommender system created with Broadway*tools observes navigations of various users and gather evaluations and annotations, to draw up a list of relevant recommendations (Web documents, keywords, etc).

Based on Jaczynski's thesis [53], different recommender systems have been developed for supporting Web browsing, but also browsing inside a Web-based information system or for query formulation in the context of a meta search engine.

5.6. Activity Modeling

5.6.1. K-MADe for Describing Human Operator or User Activities

Participant: Dominique Scapin [correspondent].

K-MADe tool (Kernel of Model for Human Activity Description Environment). The K-MADe is intended for people wishing to describe, analyze and formalize the activities of human operators, of users, in environments (computerized or not), in real or simulated situation, in the field, or in the laboratory. Although all kinds of profiles of people are possible, this environment is particularly intended for ergonomics and HCI (Human Computer Interaction) specialists. It has been developed through collaboration between ENSMA (LISI XSlaboratory) and Inria.

This year we participated in the AFIHM Working Group on Task Models (http://www.gt-mdt.fr/fr/) "Groupe de Travail de l'AFIHM sur les Modèles de Tà¢ches"). Since the early work on MAD, domain modeling task is the subject of much research (particularly in the French-speaking community), in particular the definition of formalisms and tool construction, three of which are now operational and maintained: K-MADe, eCOMM and hAMSTERS, posing an alternative to CTT. Many teams use these formalisms in a variety of goals and task models occupy a place in the field of Model Driven Engineering, and support the teaching of HCI. The WG goals are to serve as a forum between research approaches, development teams and potential users, especially for non-IT users; fostering collaboration to validate approaches; encourage feedback in teaching task models; provide the French-speaking community and eventually the international community a set of centralized, shared resources about the notion of modeling tasks.

ABSTRACTION Project-Team

5. Software and Platforms

5.1. The Apron Numerical Abstract Domain Library

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

Keywords: Convex polyhedra, Intervals, Linear equalities, Numerical abstract domain, Octagons.

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

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

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

The APRON library is developed since 2006 and currently consists of 130 000 lines of C, C++, OCaml, and Java.

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

Additionally, APRON is used internally by the team to assist the research on numeric domains and static analyses by enabling the development of fast prototypes. In 2013, APRON has been used to design a sufficient-condition generator prototype (6.3.2), the FUNCTION prototype analyzer for termination (5.6 ,6.12), a constraint solver based on numeric abstract domains (6.5), a prototype implementation and extension of the Two Variables Per Inequality abstract domain [27].

5.2. The Astrée Static Analyzer of Synchronous Software

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

Keywords: Absence of runtime error, Abstract interpretation, Static analysis, Verifier.

ASTRÉE is a static analyzer for sequential programs based on abstract interpretation [41], [31], [42], [33].

The ASTRÉE static analyzer [30], [46][1] www.astree.ens.fr aims at proving the absence of runtime errors in programs written in the C programming language.

ASTRÉE analyzes structured C programs, with complex memory usages, but without dynamic memory allocation 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.

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

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

ASTRÉE has achieved the following two unprecedented results:

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

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

5.3. The AstréeA Static Analyzer of Asynchronous Software

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

Keywords: Absence of runtime error, Abstract interpretation, Data races, Interference, Memory model, Parallel software, Static analysis, Verifier.

ASTRÉEA is a static analyzer prototype for parallel software based on abstract interpretation [43], [44], [35]. It started with support from THÉSÉE ANR project (2006–2010) and is continuing within the ASTRÉEA project (2012–2015).

The ASTRÉEA prototype www.astreea.ens.fr is a fork of the ASTRÉE static analyzer (see 5.2) that adds support for analyzing parallel embedded C software.

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.

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

ASTRÉEA is currently being applied to analyze a large industrial avionic software: 1.6 MLines of C and 15 threads, completed with a 2,500-line model of the ARINC 653 OS developed for the analysis. The analysis currently takes a few tens of hours on a 2.9 GHz 64-bit intel server using one core and generates around 1,050 alarms. The low computation time (only a few times larger than the analysis time by ASTRÉE of synchronous programs of a similar size and structure) shows the scalability of the approach (in particular, we avoid the usual combinatorial explosion associated to thread interleavings). Precision-wise, the result, while not as impressive as that of ASTRÉE, is quite encouraging. The development of AstréeA continues within the scope of the ASTRÉEA ANR project (8.1.1.2).

5.4. The MemCAD static analyzer

Participants: Xavier Rival [correspondent], Antoine Toubhans.

Keywords: Shape analysis. 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 (flat representation, graph abstraction with summaries based on inductive definitions of data-structures, such as lists) and combination operators for memory abstract domains (hierarchical abstraction, reduced product). The current implementation comes with over 200 small size test cases that are used as regression tests.

5.5. A New Tactic Engine for Coq

Participant: Arnaud Spiwack [Correspondent].

Keywords: Coq, Proof assistant, Dependent types, Tactics, Proof search.

Coq is a proof assistant based on dependent type theory developed chiefly at Inria. This project addresses longstanding usability issues when developing proofs interactively: proofs, in Coq, are typically sequences of instructions – called tactics – which transform the proof into further proof obligations. The expressiveness of tactic affects the kind of proofs which can be written realistically in Coq. Two issues have been addressed. First, providing more backtracking primitives: in a typical automated procedure – which a Coq user could write to discharge proof obligations without human effort – there is some amount of non-determinism. It is hence important to be able to devise strategies, and Coq suffered from limited options on that front. The new tactics support further primitives loosely inspired by Prolog, in particular a backtracking choice (like disjunction in Prolog), and a primitive “once” which is akind to Prolog’s soft-cut control primitive.

The second issue is more fundamental: since Coq is based on dependent types, a proof can appear in the statement of a proof obligation. As a result, tactics should be able to handle so-called dependent subgoals (where several proof obligations are left to be discharged, and the proof of one of them is mentioned into the statement of another). This was not historically the case in Coq, which had a direct influence on some users.

Both of the backtracking primitive and the dependent subgoals are part of the development version of Coq and will be part of the next release.

5.6. FuncTion: An Abstract Domain Functor for Termination

Participant: Caterina Urban.

Keywords: Conditional termination, Ranking functions, Static analysis.

FUNCTION is a research prototype static analyzer to analyze the termination of programs written in a small non-deterministic imperative language: it can infer sufficient conditions so that all executions terminate (conditional definitive termination). Following the general framework to analyze termination by abstract interpretation proposed in [40], FUNCTION infers ranking functions using piecewise-defined abstract domains. A first version of FUNCTION implemented a domain of linear ranking functions partitioned by variable bounds [24], [23]. It has been generalized in [22] and [29] to support ordinal-valued ranking functions of the form $\sum_{i=1}^{n} \omega^i f_i$ where n is a constant and each f_i is a natural-valued linear function of the variables.

The analyzer is written in OCaml and implemented on top of the APRON library (5.1). It can be used on-line through a web interface: http://www.di.ens.fr/~urban/FuncTion.html.

FUNCTION entered the 3rd Competition on Software Verification (SV-COMP 2014) in the termination category (demonstration section, no ranking).

5.7. The OpenKappa Modeling Plateform

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

Keywords: Causal traces, Model reduction, Rule-based modeling, Simulation, Static analysis.

OPENKAPPA is a collection of tools to build, debug and run models of biological pathways. It contains a compiler for the Kappa Language [53], a static analyzer [52] (for debugging models), a simulator [51], a compression tool for causal traces [50], [48], and a model reduction tool [4], [49], [58].

OPENKAPPA is developed since 2007 and, the OCaml version currently consists of 46 000 lines of OCaml. Software are available in OCaml and in Java. Moreover, an Eclipse pluggin is available.A compiler from CellDesigner into Kappa has been released in 2013.

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

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

5.8. Translation Validation

Participant: Xavier Rival [correspondent].

Keywords: Abstract interpretation, Certified compilation, Static analysis, Translation validation, Verifier.

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

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

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

5.9. Zarith

Participants: Antoine Miné [Correspondent], Xavier Leroy [Inria Paris-Rocquencourt], Pascal Cuoq [CEA LIST].

Keywords: Arbitrary precision integers, Arithmetic, OCaml.

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 an open-source project hosted at OCamlForge (http://forge.ocamlcore.org/projects/zarith) and distributed under a modified LGPL license.

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.

AOSTE Project-Team

5. Software and Platforms

5.1. TimeSquare

Participants: Charles André, Nicolas Chleq, Julien Deantoni, Frédéric Mallet [correspondant].

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 (see section 3.2), and more accurately on the associated Clock Constraint Specification Language (CCSL) for the expression of timing constraints.

TimeSquare offers four main functionalities:

1. graphical and/or textual interactive specification of logical clocks and relative constraints between them;
2. definition and handling of user-defined clock constraint libraries;
3. automated simulation of concurrent behavior traces respecting such constraints, using a Boolean solver for consistent trace extraction;
4. call-back mechanisms for the traceability of results (animation of models, display and interaction with waveform representations, generation of sequence diagrams...).

In practice TimeSquare is a plug-in developed with Eclipse modeling tools. The software is registered by the *Agence pour la Protection des Programmes*, under number IDDN.FR.001.170007.000.S.P.2009.001.10600. It can be downloaded from the site http://timesquare.inria.fr/. It has been integrated in the OpenEmbeDD ANR RNTL platform, and other such actions are under way.

5.2. K-Passa

Participants: Jean-Vivien Millo [correspondant], Robert de Simone.

This software is dedicated to the simulation, analysis, and static scheduling of Event/Marked Graphs, SDF and KRG extensions. A graphical interface allows to edit the Process Networks and their time annotations (*latency,* ...). Symbolic simulation and graph-theoretic analysis methods allow to compute and optimize static schedules, with best throughputs and minimal buffer sizes. In the case of KRG the (ultimately k-periodic) routing patterns can also be provided and transformed for optimal combination of switching and scheduling when channels are shared. KPASSA also allows for import/export of specific description formats such as UML-MARTE, to and from our other TimeSquare tool.

The tool was originally developed mainly as support for experimentations following our research results on the topic of Latency-Insensitive Design. This research was conducted and funded in part in the context of the CIM PACA initiative, with initial support from ST Microelectronics and Texas Instruments.

KPASSA is registered by the Agence pour la Protection des Programmes, under the number IDDN.FR.001.310003.000.S.P.2009.000.20700. It can be downloaded from the site http://www-sop.inria.fr/aoste/index.php?page=software/kpassa.

5.3. SynDEx

Participants: Maxence Guesdon, Yves Sorel [correspondant], Cécile Stentzel, Meriem Zidouni.

SynDEx is a system level CAD software implementing the AAA methodology for rapid prototyping and for optimizing distributed real-time embedded applications. Developed in OCaML it can be downloaded free of charge, under Inria copyright, from the general SynDEx site http://www.syndex.org.

The AAA methodology is described in section 3.3 . Accordingly, SYNDEX explores the space of possible allocations (spatial distribution and temporal scheduling), from application elements to architecture resources and services, in order to match real-time requirements; it does so by using schedulability analyses and heuristic techniques. Ultimately it generates automatically distributed real-time code running on real embedded platforms. The last major release of SYNDEX (V7) allows the specification of multi-periodic applications.

Application algorithms can be edited graphically as directed acyclic task graphs (DAG) where each edge represents a data dependence between tasks, or they may be obtained by translations from several formalisms such as Scicos (http://www.scicos.org), Signal/Polychrony (http://www.irisa.fr/espresso/Polychrony/download.php), or UML2/MARTE models (http://www.omg.org/technology/documents/profile_catalog.htm).

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.

5.4. Lopht

Participants: Thomas Carle, Manel Djemal, Zhen Zhang, Dumitru Potop Butucaru [correspondant].

The Lopht (Logical to Physical Time Compiler) has been designed as an implementation of the AAA methodology. Lopht is similar to SynDEx by relying on off-line allocation and scheduling techniques to allow real-time implementation of dataflow synchronous specifications onto multiprocessor systems. But it has two significant originality points: a stronger focus on efficiency (but without compromising correctness), and a focus on novel target architectures (many-core chips and time-triggered embedded systems).

Improved efficiency is attained through the use of classical and novel data structures and optimization algorithms pertaining to 3 fields: synchronous language compilation, classical compiler theory, and real-time scheduling. A finer representation of execution conditions allows us to make a better use of double resource reservation and thus improve latency and throughput. The use of software pipelining allows the improvement of computation throughput. The use of post-scheduling optimisations allows a reduction in the number of preemptions. The focus on novel architectures means that architecture descriptions need to define novel communication media such as the networks-on-chips (NoCs), and that real-time characteristics must include those specific to a time-triggered execution model, such as the Major Time Frame (MTF).

Significant contributions to the Lopht tool have been brought by T. Carle (the extensions concerning time-triggered platforms), M. Djemal (the extensions concerning many-core platforms), and Zhen Zhang under the supervision of D. Potop Butucaru. The tool has been used and extended during the PARSEC project. It is currently used in the IRT SystemX/FSF project, in the collaboration with Astrium Space Transportation (Airbus Defence and Space), and in the collaboration with Kalray SA. It has been developed in OCaml.

5.5. SAS

Participants: Daniel de Rauglaudre [correspondant], Yves Sorel.

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.

For a schedulable task system under EDF, when the exact preemption cost is considered, the period of the permanent phase may be much longer than the least commun multiple (LCM) of the periods of all tasks, as often found in traditional scheduling theory. Specific effort has been made to improve display in this case. The classical utilization factor, the permanent exact utilization factor, the preemption cost in the permanent phase, and the worst response time for each task are all displayed when the system is schedulable. Response times of each task relative time can also be displayed (separately).

SAS is written in OCaML, using CAMLP5 (syntactic preprocessor) and OLIBRT (a graphic toolkit under X). Both are written by Daniel de Rauglaudre. It can be downloaded from the site http://pauillac.inria.fr/~ddr/sas-dameid/.

CASCADE Project-Team (section vide)

CONTRAINTES Project-Team

5. Software and Platforms

5.1. BIOCHAM, biochemical abstract machine

Participants: François Fages, François-Marie Floch, Steven Gay, Sylvain Soliman.

The Biochemical Abstract Machine BIOCHAM is a modeling environment for systems biology distributed as open-source since 2003. Current version is v3.4, released in October. BIOCHAM uses a compositional rule-based language for modeling biochemical systems, allowing patterns for expressing set of rules in a compact form. This rule-based language is compatible with the Systems Biology Markup Language (SBML) and is interpreted with three semantics correspnding to three abstraction levels:

1. the boolean semantics (presence or absence of molecules),
2. the stochastic semantics (discrete numbers of molecules),
3. the differential semantics (concentrations of molecules).

Based on this formal framework, BIOCHAM features:

- Boolean and numerical simulators (Rosenbrock's method for the differential semantics, Gillespie's algorithm with tau lipping for the stochastic semantics);
- a temporal logic language (CTL for qualitative models and LTL(R_{lin}) with numerical constraints for quantitative models) for formalizing biological properties such as reachability, checkpoints, oscillations or stability, and checking them automatically with model-checking techniques;
- automatic search procedures to infer parameter values, initial conditions and even reaction rules from temporal logic properties;
- automatic detection of invariants, through constraint-based analysis of the underlying Petri net;
- automatic model reduction and comparison, through the use of subgraph epimorphisms [8];
- an SBGN-compatible reaction graph editor;
- an event handler allowing the encoding of hybrid models and formalisms.

BIOCHAM is implemented in GNU-Prolog and interfaced to the symbolic model checker NuSMV and to the continuous optimization tool CMAES developed by the EPI TAO.

5.2. Nicotine

Participant: Sylvain Soliman.

Nicotine is a Prolog framework dedicated to the analysis of Petri nets. It was originally built for the computation of invariants using GNU Prolog's CLP(FD) solver but has been further extended to allow import/export of various Petri nets formats. In 2013 it was ported to SWI Prolog, in order to use its more general FD solver for the satisfaction of the min-plus constraints coming from the tropical equilibration problem [13].

5.3. STSE (Spatio-Temporal Simulation Environment)

Participant: Szymon Stoma.

The overall goal of this software platform is to gather a set of open-source tools and workflows facilitating spatio-temporal simulations (preferably of biological systems) based on microscopy data. The framework currently contains modules to digitize, represent, analyze, and model spatial distributions of molecules in static and dynamic structures (e.g. growing). A strong accent is put on the experimental verification of biological models by actual, spatio-temporal data acquired using microscopy techniques. Project was initially started at Humboldt University Berlin and moved to Inria with its founder. Project webpage is: http://stse-software.org.

5.4. YeastImageToolkit

Participants: Szymon Stoma, Grégory Batt, Pascal Hersen, Artémis Llamosi.

Yeast Image Toolkit (YIT) is a set of tools facilitating segmentation and tracking of yeast cells in brightfield images. Toolkit consists of novel segmentation and tracking algorithm (CellStar), benchmark images and software allowing allowing to asses performance of the tracking. The software is currently under development and is designed to be a CellProfiler plugin. Project webpage is: http://yeast-image-toolkit.biosim.eu/.

5.5. FO-CTL(R_{lin}), first-order computation tree logic over the reals

Participants: François Fages, Thierry Martinez.

FO-CTL(R_{lin}) is a solver for full First-Order Computation Tree Logic with linear arithmetic over the reals in constrained transition systems (CTS). CTS are transition systems where both states and transitions are described with constraints. FO-CTL(R_{lin}) generalizes the implementation done in Biocham of LTL(R_{lin}) for linear traces to branching Kripke structure.

5.6. Rules2CP

Participants: François Fages, Raphaël Martin, Thierry Martinez.

Rules2CP is a rule-based modeling language for constraint programming. It is distributed since 2009 as open-source. Unlike other modeling languages for constraint programming, Rules2CP adopts a single knowledge representation paradigm based on rules without recursion, and a restricted set of data structures based on records and enumerated lists given with iterators. This allows us to model complex constraint satisfaction problems together with search strategies, where search trees are expressed by logical formulae and heuristic choice criteria are defined with preference orderings by pattern-matching on the rules' left-hand sides.

The expressiveness of Rules2CP has been illustrated in the FP6 Strep project Net-WMS by a complete library for packing problems, called PKML (Packing Knowledge Modeling Library), which, in addition to pure bin packing and bin design problems, can deal with common sense rules about weights, stability, as well as specific packing business rules.

5.7. SiLCC, linear concurrent constraint programming

Participant: Thierry Martinez.

SiLCC is an extensible modular concurrent constraint programming language relying upon linear logic. It is a complete implementation of the Linear logic Concurrent Constraint programming paradigm of Saraswat and Lincoln using the formal semantics of Fages, Ruet and Soliman. It is a single-paradigm logical language, enjoying concurrency, imperative traits, and a clean module system allowing to develop hierarchies of constraint systems within the language.

This software prototype is used to study the design of hierarchies of extensible libraries of constraint solvers. SiLCC is also considered as a possible implementation language for restructuring the code of BIOCHAM.

5.8. EMoP, existential modules for Prolog

Participant: Thierry Martinez.

EMoP is an extension of Prolog with first-class modules. These modules have the formal semantics of the LCC modules and provide Prolog with notions of namespaces, closures and objects within a simple programming model. Modules are also the support for user-definition of macros and modular syntax extensions. EMoP is bootstrapped and uses the GNU Prolog compilation chain as back-end.

5.9. CHRat, CHR with ask and tell

Participant: Thierry Martinez.

CHRat is a modular version of the well known Constraint Handling Rules language CHR, called for CHRat for CHR with *ask* and *tell*. Inspired by the LCC framework, this extension of CHR makes it possible to reuse CHRat components both in rules and guards in other CHRat components, and define hierarchies of constraint solvers. CHRat is a bootstrapped preprocessor for CHR which generates code for SWI/Prolog.

5.10. CLPGUI, constraint logic programming graphical user interface

Participant: François Fages.

CLPGUI is a generic graphical user interface written in Java for constraint logic programming. It is available for GNU-Prolog and SICStus Prolog. CLPGUI has been developed both for teaching purposes and for debugging complex programs. The graphical user interface is composed of several windows: one main console and several dynamic 2D and 3D viewers of the search tree and of finite domain variables. With CLPGUI it is possible to execute incrementally any goal, backtrack or recompute any state represented as a node in the search tree. The level of granularity for displaying the search tree is defined by annotations in the CLP program.

CLPGUI has been mainly developped in 2001 and is distributed as third-party software on GNU-Prolog and SICStus Prolog web sites. In 2009, CLPGUI has been interfaced to Rules2CP/PKML and used in the FP6 Strep Net-WMS with a non-released version.

CRYPT Team (section vide)

DEDUCTEAM Exploratory Action

5. Software and Platforms

5.1. 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 [42] for type-checking semi-full pure type systems and implements a state-of-the-art reduction machine inspired from Matita's [40] 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 its project in different files and compile them separately.

Dedukti has been developed by Mathieu Boespflug, Olivier Hermant, Quentin Carbonneaux, and Ronan Saillard. It is composed of about 1000 lines of OCaml.

5.2. Coqine, Holide and Focalide

Dedukti comes with three companion tools: Holide, an embedding of HOL proofs through the OpenTheory format [51], Coqine, an embedding of Coq proofs, and Focalide, an embedding of FoCaLiZe certified programs. All of the OpenTheory standard library and a part of Coq's and FoCaLiZe's libraries are checked by Dedukti.

A preliminary version of Coqine supports the following features of Coq: the raw Calculus of Constructions, inductive types, and fixpoint definitions. Coqine is currently being rewritten to support universes. Coqine has been developed by Mathieu Boespflug, Guillaume Burel, and Ali Assaf.

Holide supports all the features of HOL, including ML-polymorphism, constant definitions, and type definitions. It is able to translate all of the OpenTheory standard theory library. Holide has been developed by Ali Assaf.

Focalide supports the object-oriented features of FoCaLiZe, including inheritance, late-binding, redefinition and class parameters, and functional programming features of FoCaLiZe. It has been updated to work with the last version of FoCaLiZe. Focalide has been developed by Raphaël Cauderlier.

5.3. iProver Modulo

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. iProver modulo is written in OCaml, it consists of 1,200 lines of code added to the original iProver.

A tool that transforms axiomatic theories into polarized rewriting systems, thus making them usable in iProver Modulo, has also been developed. Autotheo 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 transforms the axioms into rewriting rules, and produces an input file for iProver Modulo.

iProver Modulo and autotheo have been developed by Guillaume Burel. iProver Modulo is released under a GPL license.

iProver Modulo entered CASC-24, the competition of Automated Theorem Provers held during the 24th CADE conference, in the first-order theorem division, using autotheo to orient the axioms of the problems.

5.4. Super Zenon and Zenon Modulo

Several extensions of the *Zenon* automated theorem prover (developed by Damien Doligez at *Inria* in the *Gallium* team) to Deduction modulo have been studied. These extensions intend to be applied in the context of the automatic verification of proof rules and obligations coming from industrial applications formalized using the *B* method.

The first extension, developed by Mélanie Jacquel and David Delahaye, is called *Super Zenon* and is an extension of *Zenon* to superdeduction, which can be seen as a variant of Deduction modulo. This extension is a generalization of previous experiments [10] together with Catherine Dubois and Karim Berkani (*Siemens*), where *Zenon* has been used and extended to superdeduction to deal with the *B* set theory and automatically prove proof rules of *Atelier B*. This generalization consists in allowing us to apply the extension of *Zenon* to superdeduction to any first order theory by means of a heuristic that automatically transforms axioms of the theory into rewrite rules. This work is described in [5], which also proposes a study of the possibility of recovering intuition from automated proofs using superdeduction.

The second extension, developed by Pierre Halmagrand, David Delahaye, Damien Doligez, and Olivier Hermant, is called *Zenon Modulo* and is an extension of *Zenon* to Deduction modulo. Compared to *Super Zenon*, this extension allows us to 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. To assess the approach of *Zenon Modulo*, we have applied this extension to the first order problems coming from the TPTP library. An increase of the number of proved problems has been observed, with in particular a significant increase in the category of set theory. This result in the set category allows us to be quite optimistic for the use of *Zenon Modulo* in the framework of the *BWare* project, since the *B* method is actually based on a set theory modeling technique. Over these problems of the TPTP library, we have also observed a significant proof size reduction, which confirms this aspect of Deduction modulo. These results are gathered into two publications [22], [23].

5.5. Zipperposition and Logtk

Zipperposition is an implementation of the superposition method. It experiments theory handling using its extensibility features. Current development includes splitting it into a generic library for representing logic data structures and algorithms, and a prover that uses this library. The library is called LOGTK("logic tool kit"). Zipperposition itself, in its development version, can deal with polymorphic logic, and integer and rational arithmetic. Theoretical work on an efficient inference system for arithmetic is ongoing. It entered CASC-24, the competition of Automated Theorem Provers held during the 24th CADE conference, in the first-order theorem division.

Zipperposition is developed by Simon Cruanes.

Logtk is in active development, in parallel with Zipperposition, and an unstable version is released here. The library, among other things, provides first-order terms, with polymorphic types and some type inference, first-order formulas, unification, term ordering, term rewriting, reduction of formulas to CNF, congruence closure, and an optional implementation of the meta-prover Simon Cruanes and Guillaume Burel have published about [21]. Logtk focuses on efficiency and generality of its constructs. Several term indexing structures usable for rewriting, resolution, or subsumption checking expose a functorial interface that allows to associate any data with indexed terms.

Logtk also relies on some smaller OCaml developments by Simon Cruanes, especially a full-fledged implementation of Datalog and efficient iterators.

5.6. CoLoR

CoLoR is a Coq library on rewriting theory and termination of more than 83,000 lines of code [2]. It provides definitions and theorems for:

- Mathematical structures: relations, (ordered) semi-rings.
- Data structures: lists, vectors, polynomials with multiple variables, finite multisets, matrices, finite graphs.
- Term structures: strings, algebraic terms with symbols of fixed arity, algebraic terms with varyadic symbols, pure and simply typed λ-terms.
- Transformation techniques: conversion from strings to algebraic terms, conversion from algebraic to varyadic terms, arguments filtering, rule elimination, dependency pairs, dependency graph decomposition, semantic labelling.
- Termination criteria: polynomial interpretations, multiset ordering, lexicographic ordering, first and higher order recursive path ordering, matrix interpretations.

CoLoR is distributed under the CeCILL license. It is currently developed by Frédéric Blanqui and Kim-Quyen Ly, but various people participated to its development since 2006 (see the website for more information).

5.7. HOT

HOT is an automated termination prover for higher-order rewrite systems based on the notion of computability closure and size annotation [44]. It won the 2012 competition in the category "higher-order rewriting union beta". The sources are not public. It is developed by Frédéric Blanqui.

5.8. Moca

Moca is a construction functions generator for OCaml data types with invariants.

It allows the high-level definition and automatic management of complex invariants for data types. In addition, it provides the automatic generation of maximally shared values, independently or in conjunction with the declared invariants.

A relational data type is a concrete data type that declares invariants or relations that are verified by its constructors. For each relational data type definition, Moca compiles a set of construction functions that implements the declared relations.

Moca supports two kinds of relations:

- predefined algebraic relations (such as associativity or commutativity of a binary constructor),
- user-defined rewrite rules that map some pattern of constructors and variables to some arbitrary user's define expression.

The properties that user-defined rules should satisfy (completeness, termination, and confluence of the resulting term rewriting system) must be verified by a programmer's proof before compilation. For the predefined relations, Moca generates construction functions that allow each equivalence class to be uniquely represented by their canonical value.

Moca is distributed under QPL. It is developed by Frédéric Blanqui, Pierre Weis (EPI Pomdapi) and Richard Bonichon (CEA).

5.9. Rainbow

Rainbow is a tool for automatically verifying the correctness of termination certificates expressed in the CPF XML format as used in the termination competition. Termination certificates are currently translated and checked in Coq by using the CoLoR library. But a new standalone version is under development using Coq extraction mechanism (PhD subject of Kim-Quyen Ly).

Rainbow is distributed under the CeCILL license. It is currently developed by Frédéric Blanqui and Kim-Quyen Ly. See the website for more information.

FORMES Team

5. Software and Platforms

5.1. CoLoR

Participants: Frédéric Blanqui, Kim-Quyen Ly.

CoLoR is a Coq library on rewriting theory and termination of more than 83,000 lines of code [4]. It provides definitions and theorems for:

- Mathematical structures: relations, (ordered) semi-rings.
- Data structures: lists, vectors, polynomials with multiple variables, finite multisets, matrices, finite graphs.
- Term structures: strings, algebraic terms with symbols of fixed arity, algebraic terms with varyadic symbols, pure and simply typed λ-terms.
- Transformation techniques: conversion from strings to algebraic terms, conversion from algebraic to varyadic terms, arguments filtering, rule elimination, dependency pairs, dependency graph decomposition, semantic labelling.
- Termination criteria: polynomial interpretations, multiset ordering, lexicographic ordering, first and higher order recursive path ordering, matrix interpretations.

CoLoR is distributed under the CeCILL license. It is currently developed by Frédéric Blanqui and Kim-Quyen Ly, but various people participated to its development since 2006.

5.2. HOT

Participant: Frédéric Blanqui.

HOT is an automated termination prover for higher-order rewrite systems based on the notion of computability closure and size annotation [24]. It won the 2012 competition in the category "higher-order rewriting union beta". The sources are not public.

5.3. Moca

Participant: Frédéric Blanqui.

Moca is a construction functions generator for OCaml data types with invariants.

It allows the high-level definition and automatic management of complex invariants for data types. In addition, it provides the automatic generation of maximally shared values, independently or in conjunction with the declared invariants.

A relational data type is a concrete data type that declares invariants or relations that are verified by its constructors. For each relational data type definition, Moca compiles a set of construction functions that implements the declared relations.

Moca supports two kinds of relations:

- predefined algebraic relations (such as associativity or commutativity of a binary constructor),
- user-defined rewrite rules that map some pattern of constructors and variables to some arbitrary users defined expression.

The properties that user-defined rules should satisfy (completeness, termination, and confluence of the resulting term rewriting system) must be verified by a programmer's proof before compilation. For the predefined relations, Moca generates construction functions that allow each equivalence class to be uniquely represented by their canonical value.

Moca is distributed under QPL. It is developed by Frédéric Blanqui, Pierre Weis (EPI Pomdapi) and Richard Bonichon (CEA).

5.4. Rainbow

Participants: Frédéric Blanqui, Kim-Quyen Ly.

Rainbow is a tool for automatically verifying the correctness of termination certificates expressed in the CPF XML format as used in the termination competition. Termination certificates are currently translated and checked in Coq by using the CoLoR library. But a new standalone version is under development using Coq extraction mechanism (PhD subject of Kim-Quyen Ly).

Rainbow is distributed under the CeCILL license. It is currently developed by Frédéric Blanqui and Kim-Quyen Ly. See the web site for more information.

5.5. CoqMT

Participants: Qian Wang [correspondant], Jean-Pierre Jouannaud.

The proof-assistant Coq is based on a complex type theory, which resulted from various extensions of the Calculus of Constructions studied independently from each other. With the collaboration of Bruno Barras, we decided to address the challenge of proving the real type theory underlying Coq, and even, indeed, of its recent extension CoqMT developed in FORMES by Pierre-Yves Strub. To this end, we have studied formally the theory CoqMTU, which extends the pure Calculus of Constructions by inductive types, a predicative hierarchy of universes, and a decidable theory T for some first-order inductive types. Recently, we were able to announce the complete certification of CoqMTU in Coq augmented with appropriate intuitionistic set-theoretic axioms in order to fight Gödel's incompleteness theorem~[16]. As a consequence, Coq and CoqMTU are the first proof assistants, of which consistency (relative to intuitionistic set theory IZF augmented with the afore-mentioned axioms) is formally entirely proved (in Coq). While previous formal proofs for Coq and other proof assistants all assumed strong normalization, the present one *proves* strong normalization thanks to the new notion of *strongly-normalizing* model introduced by Bruno Barras. While consistency is done already, decidability of type-checking in CoqMTU remains to be done. This is a straightforward consequence for Coq, but a non-trivial task for CoqMTU because of the interaction between inductive types and the first-order theory T. It should however be done by the summer of 2014. We consider this work as a major scientific achievement of the team.

5.6. SimSoC

Participants: Vania Joloboff [correspondant], Antoine Rouquette, Shenpeng Wang.

SimSoC is an infrastructure to run simulation models which comes along with a library of simulation models. **SimSoC** allows its users to experiment various system architectures, study hardware/software partition, and develop embedded software in a co-design environment before the hardware is ready to be used. **SimSoC** aims at providing high performance, yet accurate simulation, and provide tools to evaluate performance and functional or non functional properties of the simulated system.

SimSoC is based on SystemC standard and uses Transaction Level Modeling for interactions between the simulation models. The current version is based on the open source libraries from the OSCI Consortium: SystemC version 2.3 and TLM 2.0.1 [39], [21]. Hardware components are modeled as TLM models, and since TLM is itself based on SystemC, the simulation is driven by the SystemC kernel. We use standard, unmodified, SystemC, hence the simulator has a single simulation loop.

The third open source version of **SimSoC**, release 0.8.0, has been released in September 2013. It contains a full simulator for ARM (V5 and V6) and PowerPC both running at an average speed of about 100 Millions instructions per second in, and a deprecated simulator for the MIPS architecture. **SimSoC** is distributed under LGPL on Inria Gforge web site.

5.7. SimSoC-Cert

Participants: Frédéric Blanqui, Vania Joloboff, Jean-François Monin [correspondant], Xiaomu Shi.

Simulators such as **SimSoC** make it possible to reduce development time and development cost, allowing for the software engineers to run fast iterative cycles without requiring a hardware development board. Then a critical issue is: *does the simulator actually simulate the real hardware?*

Considering only one module in **SimSoC**, namely the ARM simulator, it somehow encodes the 1138 pages of the ARM reference manual in C++. The whole simulator, which simulates ARM and PowerPC architecture, includes about 60,000 lines of manually coded C++ code. Then, mistakes in the hand written code are unavoidable and difficult to find due to the complexity. From the experiments performed on **SimSoC**, bugs bringing a wrong behavior were observed from time to time but it was hard to reveal where they were. Using intensive tests can cover most of the instructions, but still left some untested rare cases of instructions, which lead to potential problems.

Therefore, a better approach is required to gain confidence in the correctness of the simulator. Our proposal has been to certify the ARM CPU simulator from **SimSoC** using formal methods. We aimed at proving a significant part of the correctness of **SimSoC** in order to support the claim that the implementation of the simulator and the real hardware system will exhibit the same behavior.

In addition, we developed tools that can automatically generate in various C the core simulator, including the decoding functions and the instruction set of the ARMv6 architecture manual [18] (implemented by the ARM11 processor family). The input of SimSoC-Cert is the ARMv6 architecture manual itself.

In order to get the required flexibility and accuracy, we wanted to experiment a direct approach based on a general proof assistant such as Coq. Fortunately, an operational semantics formalized in Coq of a large enough subset of the C language is available from the **CompCert** project. We then decided to base our correctness proofs on this technology. Up to our knowledge, this is the first development of formal correctness proofs based on operational semantics, at least at this scale.

Based on this, we first developed *simlight* (8000 generated lines of C, plus 1500 hand-written lines of C), a simulator for ARMv6 programs using no peripheral and no coprocessor. Next, we developed *simlight2*, a fast ARMv6 simulator integrated inside a SystemC/TLM module, now part of SimSoC v0.8.

We can also generate similar programs for SH4 [20] but this is still experimental (work done by Frédéric Tuong in 2011).

Finally, we proved that the C code for simulating ARM instructions in Simlight is correct with respect to the Coq model.

GALLIUM Project-Team

5. Software and Platforms

5.1. OCaml

Participants: Damien Doligez [correspondant], Alain Frisch [LexiFi], Jacques Garrigue [Nagoya University], Fabrice Le Fessant, Xavier Leroy, Luc Maranget.

OCaml, formerly known as Objective Caml, is our flagship implementation of the Caml language. From a language standpoint, it extends the core Caml language with a fully-fledged object and class layer, as well as a powerful module system, all joined together by a sound, polymorphic type system featuring type inference. The OCaml system is an industrial-strength implementation of this language, featuring a high-performance native-code compiler for several processor architectures (IA32, AMD64, PowerPC, ARM, etc) as well as a bytecode compiler and interactive loop for quick development and portability. The OCaml distribution includes a standard library and a number of programming tools: replay debugger, lexer and parser generators, documentation generator, and compilation manager.

Web site: http://caml.inria.fr/

5.2. CompCert C

Participants: Xavier Leroy [correspondant], Sandrine Blazy [EPI Celtique], Jacques-Henri Jourdan.

The CompCert C verified compiler is a compiler for a large subset of the C programming language that generates code for the PowerPC, ARM and x86 processors. The distinguishing feature of Compcert is that it has been formally verified using the Coq proof assistant: the generated assembly code is formally guaranteed to behave as prescribed by the semantics of the source C code. The subset of C supported is quite large, including all C types except `long double`, all C operators, almost all control structures (the only exception is unstructured `switch`), and the full power of functions (including function pointers and recursive functions but not variadic functions). The generated PowerPC code runs 2–3 times faster than that generated by GCC without optimizations, and only 7% (resp. 12%) slower than GCC at optimization level 1 (resp. 2).

Web site: http://compcert.inria.fr/

5.3. The diy tool suite

Participants: Luc Maranget [correspondant], Jade Alglave [University College London], Susmit Sarkar [University of St Andrews], Peter Sewell [University of Cambridge].

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

The tool suite and a comprehensive documentation are available from http://diy.inria.fr/.

5.4. Zenon

Participant: Damien Doligez.

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).

Web site: http://zenon-prover.org/

5.5. JoCaml

Participant: Luc Maranget.

JoCaml is an experimental extension of OCaml that adds support for concurrent and distributed programming, following the programming model of the join-calculus.

Web site: http://jocaml.inria.fr/

MUTANT Project-Team

5. Software and Platforms

5.1. Antescofo

Participants: Arshia Cont, Jean-Louis Giavitto, Florent Jacquemard, José Echeveste.

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.

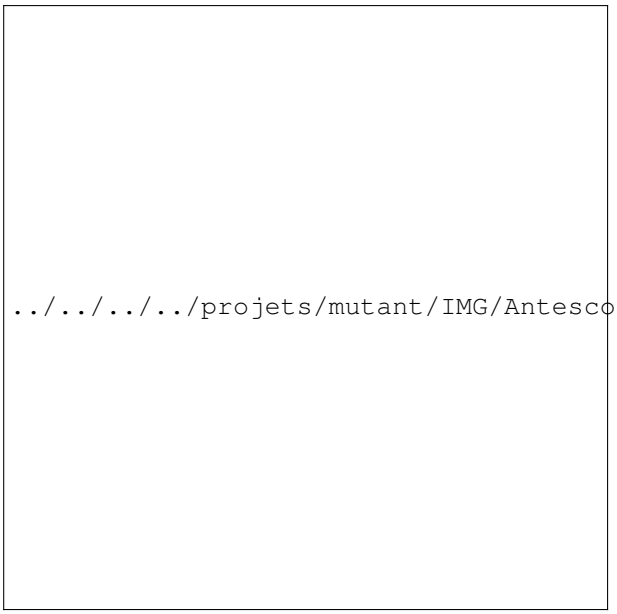

Figure 5. General scheme of Antescofo virtual machine

Antescofo is developed as modules for *Max* and *PureData* real-time programming environments.

A complete new version of Antescofo has been released on November 2013 on Ircam Forumnet. This version is the result of one year of intensive effort by MuTant team members and associate artists.

This release include major improvements on the reactive language: richer set of synchronization strategies and control structures, dynamic continuous actions, high order function and processes, historicized variables and dynamic expressions everywhere (delays or periods as variables or expressions).

The new internal architecture unifies completely the handling of external (musical) events and the handling of internal (logical) events in a framework able to manage multiple time frames (relative, absolute or computed).

The new version targets the Max and PureData (Pd) environments on Mac, but also on Linux (Pd version) and offers also a standalone offline version. The standalone version is used to simulate a performance in Ascograph.

An important enhancement has been made by proposing a richer set of synchronization strategies between the event recognized by the listening machine and the action to be performed by the reactive engines. Theses new strategies include *anticipative* strategies that exhibits a smoother musical behavior. These strategies are now tested in various musical situations like accompaniments and in the creation of new pieces.

Some new results including a behavioral semantics of the static kernel of the Antescofo reactive engine [17] and tools for formal verification and conformance testing of the system [24], [35] are presented below.

5.2. Ascograph: Antescofo Visual Editor

Participants: Thomas Coffy [ADT], Arshia Cont, José Echeveste.

The Antescofo programming language can be extended to visual programing to better integrate existing scores and to allow users to construct complex and embedded temporal structures that are not easily integrated into text. This project is held since October 2012 thanks to Inria ADT Support.

AscoGraph, the new Antescofo graphical score editor has been released this year. It 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 (see Figure 3). 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.

One other really important feature is the score import from MusicXML or MIDI files, which make the complete workflow of the composition of a musical piece much easier than before.

AscoGraph is strongly connected with Antescofo core object (using OSC over UDP): when a score is edited and modified it is automatically reloaded in Antescofo, and on the other hand, when Antescofo follows a score (during a concert or rehearsal) both graphical and textual view of the score will scroll and show the current position of Antescofo.

AscoGraph is released under Open-Source MIT license and has been released publicly along with new Antescofo architecture during IRCAM Forum 2013.

PARKAS Project-Team

5. Software and Platforms

5.1. Lucid Synchrone

Participant: Marc Pouzet [contact].

Synchronous languages, type and clock inference, causality analysis, compilation

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.

It is distributed under binary form, at URL http://www.di.ens.fr/~pouzet/lucid-synchrone/.

The language was used, from 1996 to 2006 as a laboratory to experiment various extensions of the language Lustre. Several programming constructs (e.g. merge, last, mix of data-flow and control-structures like automata), type-based program analysis (e.g., typing, clock calculus) and compilation methods, originaly introduced in Lucid Synchrone are now integrated in the new SCADE 6 compiler developped at Esterel-Technologies and commercialized since 2008.

Three major release of the language has been done and the current version is V3 (dev. in 2006). As of 2013, the language is still used for teaching and in our research but we do not develop it anymore. Nonetheless, we have integrated several features from Lucid Synchrone in new research prototypes described below. The Heptagon language and compiler are a direct descendent of it. The new language Zélus for hybrid systems modeling borrows many features originaly introduced in Lucid Synchrone.

5.2. ReactiveML

Participants: Guillaume Baudart, Louis Mandel [contact], Cédric Pasteur.

Programming language, synchronous reactive programming, concurrent systems, dedicated type-systems.

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.

ReactiveML is distributed at URL http://reactiveml.org. The compiler is distributed under the terms of the Q Public License and the library is distributed under the terms of the GNU Library General Public License. The development of ReactiveML started at the University Paris 6 (from 2002 to 2006).

The language was mainly used for the simulation of mobile ad hoc networks at the Pierre and Marie Curie University and for the simulation of sensor networks at France Telecom and Verimag (CNRS, Grenoble). A new application to mixed music programming has been developed.

In 2013, a new web site has been developed. New programming constructs have been added. The runtime system has been cleanup. Moreover, a new implementation based on the PhD of Cédric Pasteur has also been provided http://reactiveml.org/these_pasteur.

5.3. Heptagon

Participants: Cédric Pasteur [contact], Brice Gelineau, Léonard Gérard, Adrien Guatto, Marc Pouzet.

Synchronous languages, compilation, optimizing compilation, parallel code generation, behavioral synthesis.

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.

Some extensions have already been made, most notably automata, a parallel code generator with Futures, support for correct and efficient in-place array computations. It's currently used to experiment with linear typing for arrays and also to introduce a concept of asynchronous parallel computations. The compiler developed in our team generates C, C++, java and VHDL code.

Transfer activities based on our experience in Heptagon are taking place through the "Fiabilité and Sûreté de Fonctionnement" project at IRT SystemX, led by Alstom Transport, since 2013.

Heptagon is jointly developed with Gwenael Delaval and Alain Girault from the Inria POP ART team (Grenoble). Gwenael Delaval is developing the controller synthesis tool BZR (http://bzr.inria.fr/) above Heptagon. Both software are distributed under a GPL licence.

5.4. Lucy-n: an n-synchronous data-flow programming language

Participants: Albert Cohen, Louis Mandel [contact], Adrien Guatto, Marc Pouzet.

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.

The language compiler and associated tools are available in a binary form at http://www.lri.fr/~mandel/lucy-n.

In 2013, a complete re-implemtantion has been started. This new version will take into account the new features developed during the PhD of Adrien Guatto. Parallel code generation for this new version also involves compilation and runtime system research in collaboration with Nhat Minh Lê and Robin Morisset.

5.5. ML-Sundials

Participants: Timothy Bourke, Jun Inoue, Marc Pouzet [contact].

The ML-Sundials bindings allow the use of the state-of-the-art Sundials numerical simulation library from OCaml programs (like, for instance, the Zélus runtime). The Sundials packages includes three main components: CVODE, IDA, and KINSOL.

This year we redesigned and reimplemented the interface to CVODE to fix a problem with memory leaks between OCaml and C heaps. We have submitted an APP request for this code. The CVODE component is an important part of our work on the Zélus programming language.

We also developed a new interface for the IDA component, which we have started to use in our experiments with DAEs (Modelica).

We plan to develop an interface for the remaining KINSOL component over the next three months and then to release the entire library under an open-source license.

5.6. Zélus

Participants: Timothy Bourke, Marc Pouzet [contact].

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 behabiors. 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.

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.

This year we reimplemented several basic numeric solver algorithms after a careful analysis of the Simulink versions together with the binding to SUNDIALS CVODE. This was necessary to enable detailed comparsions between our tool and Simulink (the de facto industrial standard in this domain). We also improved the algorithm for zero-crossing detection, simplified and streamlined the back-end interface.

We developed several new examples to aid in the development, debugging, and dissemination of our work together with various talks and demonstrations. These included a simple backhoe model (which served as a introducing example in the HSCC paper [12]), an adaptive control example from Astrom and Wittenmark's text, and a model of Zeno behaviour based on a zig-zagging object (presented at Synchron).

Zélus has been released officially in 2013 with several complete documented examples on http://zelus.di.ens.fr. An important software development has been done in the compiler internals during year 2013: a new *causality analysis* has been designed and implemented and a new back-end to generate efficient sequential code for both the discrete step and the continuous step.

5.7. GCC

Participants: Albert Cohen [contact], Tobias Grosser, Antoniu Pop, Feng Li, Riyadh Baghdadi, Nhat Minh Lê.

Compilation, optimizing compilation, parallel data-flow programming automatic parallelization, polyhedral compilation. http://gcc.gnu.org

Licence: GPLv3+ and LGPLv3+

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.

PARKAS contributes to the polyhedral compilation framework, also known as Graphite. We also distribute an experimental branch for a stream-programming extension of OpenMP called OpenStream (used in numerous research activites and grants). This effort borrows key design elements to synchronous data-flow languages.

Tobias Grosser is one of main contributors of the Graphite optimization pass of GCC.

5.8. isl

Participants: Sven Verdoolaege [contact], Tobias Grosser, Albert Cohen.

Presburger arithmetic, integer linear programming, polyhedral library, automatic parallelization, polyhedral compilation. http://freshmeat.net/projects/isl

Licence: MIT

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.

isl has become the de-facto standard for every recent polyhedral compilation project. Thanks to a license change from LGPL to MIT, its adoption is also picking up in industry.

5.9. ppcg

Participants: Sven Verdoolaege [contact], Tobias Grosser, Riyadh Baghdadi, Albert Cohen.

Presburger arithmetic, integer linear programming, polyhedral library, automatic parallelization, polyhedral compilation. http://freshmeat.net/projects/ppcg

Licence: MIT

More tools are being developed, based on isl. 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).

5.10. Tool support for the working semanticist

Participant: Francesco Zappa Nardelli [contact].

Languages, semantics, tool support, theorem prouvers.

We are working on tools to support large scale semantic definitions, for programming languages and architecture specifications. For that we develop two complementary tools, Ott and Lem.

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

1. a LaTeX source file that defines commands to build a typeset version of the definition;
2. a Coq version of the definition;
3. an Isabelle version of the definition; and
4. a HOL version of the definition.

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

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

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

In collaboration with Peter Sewell (Cambridge University) and Scott Owens (University of Kent).

The current version of Ott is about 30000 lines of OCaml. The tool is available from http://moscova.inria.fr/~zappa/software/ott (BSD licence). It is widely used in the scientific community. In 2013 we implemented several bug-fixes, we kept the theorem prouver backends up-to date with the prover evolution, and we have been working toward a closer integration with the Lem tool.

The development version of Lem is available from http://www.cs.kent.ac.uk/people/staff/sao/lem/.

5.11. Cmmtest: a tool for hunting concurrency compiler bugs

Participants: Francesco Zappa Nardelli [contact], Robin Morisset, Pankaj More, Anirudh Kumar, Pankaj Prateek Kewalramani, Pejman Attar.

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.

PI.R2 Project-Team

4. Software and Platforms

4.1. COQ (http://coq.inria.fr)

Participants: Bruno Barras [Inria Saclay], Yves Bertot [Marelle team, Sophia], Pierre Boutillier, Xavier Clerc [SED team], Pierre Courtieu [CNAM], Maxime Dénès [Marelle team, Sophia], Julien Forest [CNAM], Stéphane Glondu [CARAMEL team, Nancy Grand Est], Benjamin Grégoire [Marelle team, Sophia], Vincent Gross [Consultant at NBS Systems], Hugo Herbelin [correspondant], Pierre Letouzey, Assia Mahboubi [SpecFun team, Saclay], Julien Narboux [University of Strasbourg], Jean-Marc Notin [Ecole Polytechnique], Christine Paulin [Proval team, Saclay], Pierre-Marie Pédrot, Loïc Pottier [Marelle team, Sophia], Matthias Puech, Yann Régis-Gianas, François Ripault, Matthieu Sozeau, Arnaud Spiwack [Abstraction team, ENS], Pierre-Yves Strub [IMDEA, Madrid], Enrico Tassi [SpecFun team, Saclay], Benjamin Werner [Ecole Polytechnique].

4.1.1. Version 8.5

Version 8.5 is expected to be released after the summer of 2014. It will be a major release of the Coq proof assistant, including 6 major new features:

- Parallel development and compilation, inside files and across files, by Enrico Tassi (Inria SpecFun), a result of the Paral-ITP ANR project.
- New proof engine of Arnaud Spiwack (formerly pi.r2 postdoc), more expressive and with clearer semantics.
- Native compilation by Maxime Dénès and Benjamin Grégoire (Inria Marelle, M. Dénès is now at the University of Pennsylvania). A compilation scheme from Coq to OCaml to native code, considerably improving on the previous virtual machine implementation by B. Grégoire.
- A Universe Polymorphic extension by Matthieu Sozeau that allows universe-generic developments, as required by the Homotopy Type Theory library for example.
- Primitive projections for records by Matthieu Sozeau.
- A new document generation system by F. Ripault and Yann Régis-Gianas.

A more detailed description of all the new features will be made in next year's report, but some elements can already be found below.

4.1.2. Evaluation algorithms

Pierre Boutillier has worked on the practical implementation of the unfolding algorithm for global constants that he proposed last year, so that it could become the default process to simplify terms by tactics. A formal presentation has been written in his PhD, to be defended in February 2014.

4.1.3. Internal representation of projections

The change of representation for record projections implemented by Matthieu Sozeau will be part of the 8.5 release. It provides not only exponential gains in performance (type-checking and comparison time and space usage) but also a better basis to work with canonical structures in the unification algorithm, allowing to improve for example the ssreflect inference mechanism significantly. Benchmarks on the HoTT library and a groupoid model construction confirm the exponential gain in performance.

4.1.4. Universes

Matthieu Sozeau followed up his work on universe polymorphism and uncovered important theoretical problems regarding conversion and unification of universe polymorphic constants in the presence of cumulativity and the Prop $\leq$ Type rule. After a careful study of the alternative solutions, he designed a practical correction for the issue and developed a paper proof of conservativity of the complete new system over the original theory of Coq. A paper describing this work has been submitted. The universe polymorphic system, already in use by the HoTT community, will be part of the upcoming 8.5 release.

4.1.5. 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 formalization of the normalization proof of a simply-typed lambda calculus. Wojciech Jedynak, a student of Dariusz Biernacki and Małgorzata Biernacka at the University of Wroclaw, is working under his supervision to do the remaining work before an official release of the plugin can be done. Wojciech Jedynak applied for a 4-month internship supervised by Matthieu Sozeau on an extension of Equations, to start in March 2014.

4.1.6. Internal architecture of the Coq software

With the help of many others, Pierre Letouzey organized in November 2013 the migration of the official Coq source repository from subversion to git. The native use of this decentralized version control system eases the exchange of code amongst Coq developpers (either from the Coq dev team or from external contributors).

Pierre Letouzey, Pierre-Marie Pédrot and Xavier Clerc have continued to work at improving the quality of the OCaml code which composes Coq :

- Many modules have been revised, in particular with cleaner naming convention.
- Almost all uses of the generic OCaml comparison has been chased and transformed into specific code. This avoided many potential bugs with advanced structures, while improving performances at the same time.
- The codes handling OCaml exceptions have been reworked to avoid undue interceptions of critical exceptions.
- Issues involving exceptions are now quite simpler to debug, thanks to easy-to-obtain backtraces.

4.1.7. Efficiency

Pierre-Marie Pédrot has been working on the overall optimization of Coq, by tracking hotspots in the code. Coq trunk 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.

Pierre Letouzey has improved the representation of Coq binary files : these files are now smaller (thanks to more sharing), and are reloaded quickly by Coq (thanks to deferred loading of opaque proof terms, which are large and almost never accessed by the user).

4.1.8. Documentation generation

François Ripault and Yann Régis-Gianas developed a new version of coqdoc, the documentation generator of Coq. This new implementation is based on the interaction protocol with the Coq system and should be more robust with respect to the evolution of Coq.

4.1.9. General maintenance

Pierre Letouzey has been the main maintainer of Coq with extra contributions from Hugo Herbelin, Pierre Boutillier, Matthieu Sozeau, Pierre-Marie Pédrot, ...

4.1.10. Modules in Coq

In 2013, Pierre Letouzey has proposed an important rework of the code implementing the module system of Coq. This code was inherited from the successive works of several PhD students, and was in pretty poor shape. While being equivalent in terms of features for the user, the new code should be quite more readable and robust, as well as more efficient: the memory sharing of modular structures should be better, leading to reduced memory footprint for Coq as well as smaller Coq compiled files.

4.1.11. The Coq extraction

Pierre Letouzey has collaborated with colleagues with the aim of extending the extraction tool to additional target languages:

- `C++` with Gabriel Dos Reis and his student Robert Schumacher
- `F#` with David Monniaux

These experiments have been quite promising. In the case of `C++`, an article has been written, it should be re-submitted soon for publication.

4.1.12. Formalisation in Coq

Hugo Herbelin's type-theoretic construction of semi-simplicial sets [22] has been formalised in Coq.

Matthieu Sozeau and Nicolas Tabareau formalised a groupoid model in Coq http://github.com/mattam82/groupoid.

Jaime Gaspar has verified in Coq the correctness of Jean-Louis Krivine's proof that Zermelo-Fraenkel set theory without choice ZF is contained in a variant ZF_ε (useful for Krivine's classical realisability).

4.2. Other software developments

In collaboration with François Pottier (Inria Gallium), Yann Régis-Gianas maintained Menhir, an LR parser generator for OCaml.

Yann Régis-Gianas started the development of the "Hacking Dojo", a web platform to automatically grade programming exercises.

In a different vein, Jaime Gaspar showed that is not always possible to get cross-references right in LaTeX by presenting a LaTeX file where a cross-reference is always wrong (no matter how many times we compile the file).

POLSYS Project-Team

5. Software and Platforms

5.1. FGb

Participant: Jean-Charles Faugère [contact].

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.
See also the web page http://www-polsys.lip6.fr/~jcf/Software/FGb/index.html.

- ACM: I.1.2 Algebraic algorithms
- Programming language: C/C++

5.2. RAGlib

Participant: Mohab Safey El Din [contact].

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

5.3. Epsilon

Participant: Dongming Wang [contact].

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

PROSECCO Project-Team

5. Software and Platforms

5.1. ProVerif

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

PROVERIF (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 proverif.inria.fr, under the GPL license.

5.2. CryptoVerif

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

CRYPTOVERIF(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 cryptoverif.inria.fr, under the CeCILL license.

5.3. Cryptosense Analyzer

Participants: Graham Steel [correspondant], Romain Bardou.

See also the web page http://cryptosense.com.

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

In June 2013 we submitted a patent application (13 55374) on the reverse engineering process. We also concluded a license agreement between Inria PROSECCO and the nascent spin-off company Cryptosense to commercialize the tool.

5.4. miTLS

Participants: Alfredo Pironti [correspondant], Karthikeyan Bhargavan, Cedric Fournet [Microsoft Research], Pierre-Yves Strub [IMDEA], Markulf Kohlweiss [Microsoft Research].

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, and two updates to the software were released in 2013. The software and associated research materials are available from http://mitls.rocq.inria.fr.

5.5. WebSpi

Participants: Karthikeyan Bhargavan [correspondant], Sergio Maffeis [Imperial College London], Chetan Bansal [BITS Pilani-Goa], Antoine Delignat-Lavaud.

WebSpi is a library that aims to make it easy to develop models of web security mechanisms and protocols and verify them using ProVerif. It captures common modeling idioms (such as principals and dynamic compromise) and defines a customizable attacker model using a set of flags. It defines an attacker API that is designed to make it easy to extract concrete attacks from ProVerif counterexamples.

WebSpi has been used to analyze social sign-on and social sharing services offered by prominent social networks, such as Facebook, Twitter, and Google, on the basis of new open standards such as the OAuth 2.0 authorization protocol.

WebSpi has also been used to investigate the security of a number of cryptographi web applications, including password managers, cloud storage providers, an e-voting website and a conference management system.

WebSpi is under development and released as an open source library at http://prosecco.inria.fr/webspi/

5.6. Defensive JavaScript

Participants: Antoine Delignat-Lavaud [correspondant], Karthikeyan Bhargavan, Sergio Maffeis [Imperial College London].

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. DJS is especially useful to write security APIs that can be loaded in untrusted pages, for instance an OAuth library such as the one used by "Login with Facebook". It is also useful to write secure host-proof web applications, and more generally for cryptography that happens on the browser.

The DJS type checker and various libraries written in DJS are available from http://www.defensivejs.com.

SECRET Project-Team (section vide)

CAD Team (section vide)

CLASSIC Project-Team (section vide)

GAMMA3 Project-Team

3. Software and Platforms

3.1. BLGEOL-V1 software

Participants: Patrick Laug [correspondant], Houman Borouchaki.

BLGEOL-V1 software can generate hex-dominant meshes of geologic structures complying with different geometric constraints: surface topography (valleys, reliefs, rivers), geologic layers and underground workings. First, a reference 2D domain is obtained by projecting all the line constraints into a horizontal plane. Different size specifications are given for rivers, outcrop lines and workings. Using an adaptive methodology, the size variation is bounded by a specified threshold in order to obtain a high quality quad-dominant mesh. Secondly, a hex-dominant mesh of the geological medium is generated by a vertical extrusion, taking into account the surfaces found (interfaces between two layers, top or bottom faces of underground workings). The generation of volume elements follows a global order established on the whole set of surfaces to ensure the conformity of the resulting mesh.

MATHRISK Project-Team

5. Software and Platforms

5.1. PREMIA

Participants: Antonino Zanette, Mathrisk Research Team, Agnès Sulem [correspondant].

Premia is a software designed for option pricing, hedging and financial model calibration. It is provided with it's C/C++ source code and an extensive scientific documentation. https://www-rocq.inria.fr/mathfi/Premia

The Premia project keeps track of the most recent advances in the field of computational finance in a well-documented way. It focuses on the implementation of numerical analysis techniques for both probabilistic and deterministic numerical methods. An important feature of the platform Premia is the detailed documentation which provides extended references in option pricing.

Premia is thus a powerful tool to assist Research & Development professional teams in their day-to-day duty. It is also a useful support for academics who wish to perform tests on new algorithms or pricing methods without starting from scratch.

Besides being a single entry point for accessible overviews and basic implementations of various numerical methods, the aim of the Premia project is:

1. to be a powerful testing platform for comparing different numerical methods between each other;
2. to build a link between professional financial teams and academic researchers;
3. to provide a useful teaching support for Master and PhD students in mathematical finance.

- AMS: 91B28;65Cxx;65Fxx;65Lxx;65Pxx
- License: Licence Propriétaire (genuine license for the Consortium Premia)
- Type of human computer interaction: Console, interface in Nsp, Web interface
- OS/Middelware: Linux, Mac OS X, Windows
- APP: The development of Premia started in 1999 and 15 are released up to now and registered at the APP agency.
- Programming language: C/C++ librairie Gtk
- Documentation: the PNL library is interfaced via doxygen
- Size of the software: 280580 lines for the Src part only, that is 11 Mbyte of code, 130400 lines for PNL, 103 Mbyte of PDF files of documentation.
- interfaces : Nsp for Windows/Linux/Mac, Excel, binding Python, and a Web interface.
- Publications: [1], [68], [75], [83], [86], [55]

5.1.1. Content of Premia

Premia contains various numerical algorithms (Finite-differences, trees and Monte-Carlo) for pricing vanilla and exotic options on equities, interest rate, credit and energy derivatives.

1. **Equity derivatives:**

   The following models are considered:

   Black-Scholes model (up to dimension 10), stochastic volatility models (Hull-White, Heston, Fouque-Papanicolaou-Sircar), models with jumps (Merton, Kou, Tempered stable processes, Variance gamma, Normal inverse Gaussian), Bates model.

   For high dimensional American options, Premia provides the most recent Monte-Carlo algorithms: Longstaff-Schwartz, Barraquand-Martineau, Tsitsklis-Van Roy, Broadie-Glassermann, quantization methods and Malliavin calculus based methods.

Dynamic Hedging for Black-Scholes and jump models is available.

Calibration algorithms for some models with jumps, local volatility and stochastic volatility are implemented.

2. **Interest rate derivatives**

   The following models are considered:

   HJM and Libor Market Models (LMM): affine models, Hull-White, CIR++, Black-Karasinsky, Squared-Gaussian, Li-Ritchken-Sankarasubramanian, Bhar-Chiarella, Jump diffusion LMM, Markov functional LMM, LMM with stochastic volatility.

   Premia provides a calibration toolbox for Libor Market model using a database of swaptions and caps implied volatilities.

3. **Credit derivatives: CDS, CDO**

   Reduced form models and copula models are considered.

   Premia provides a toolbox for pricing CDOs using the most recent algorithms (Hull-White, Laurent-Gregory, El Karoui-Jiao, Yang-Zhang, Schönbucher)

4. **Hybrid products**

   PDE solver for pricing derivatives on hybrid products like options on inflation and interest or change rates is implemented.

5. **Energy derivatives: swing options**

   Mean reverting and jump models are considered.

   Premia provides a toolbox for pricing swing options using finite differences, Monte-Carlo Malliavin-based approach and quantization algorithms.

5.1.2. Premia design

Premia has managed to grow up over a period of more than a dozen years; this has been possible only because contributing an algorithm to Premia is subject to strict rules, which have become too stringent. To facilitate contributions, a standardized numerical library (PNL) has been developed by J. Lelong under the LGPL since 2009, which offers a wide variety of high level numerical methods for dealing with linear algebra, numerical integration, optimization, random number generators, Fourier and Laplace transforms, and much more. Everyone who wishes to contribute is encouraged to base its code on PNL and providing such a unified numerical library has considerably eased the development of new algorithms which have become over the releases more and more sophisticated. An effort will be made to continue and stabilize the development of PNL.

1. Development of the PNL. Here are the major 2013 contributions (by Jérôme Lelong):
   1. PNL relies on CMake for compiling.
   2. Add the sampling of new distributions: log-normal, inverse Gaussian, asymmetric double exponential distributions.
   3. Add the computation of eigenvalues and eigenvectors for complex matrices. Based on this new function, add the computation of the matrix logarithm for complex matrices.
   4. Add Newton's algorithm with Armijo line search.
   5. The top level PnlOjbect is modified to keep track of the number of references on an object to improve memory management in lists. This delicate change in the core of the library enabled us to speed codes based on lists by a great deal.
   6. Several other functions have also been added.
2. Premia

1. The compilation of Premia is now based on CMake which is a cross-platform building tool. It allows us to maintain a single building chain and to automatically generate Makefiles or a Visual project. This technology change significantly improves our ability to generate Windows versions.
2. Add support for PnlMatrix both in Premia VAR and in the Nsp toolbox.
3. A model size change in the Nsp GUI automatically propagates to all parameters thanks to the addition a *Return* callback in the GUI.
4. Some fixes in the core of Premia: several setters were broken.
5. Refactor the credit toolbox to simplify the number of products.
6. Scripts to generate new model templates have been significantly improved and reimplemented in Python.
7. Improve the generic functions `Get`, `FGet`, `Show`, `PrintVar` and `FScanVar` to enable all the models to use them. This led us to remove a lot of code.

5.1.3. Algorithms implemented in Premia in 2013

Premia 15 was delivered to the consortium members in March 2013. It contains the following new algorithms:

- **Interest Rate, Inflation, FX**
  - Inflation products with stochastic volatility and stochastic interest rates. S. Singor, L. Grzelak C.W.Oosterlee D.D.B. van Bragt.
  - On cross-currency models with stochastic volatility and correlated interest rates. L. Grzelak C.W.Oosterlee. *Applied Math. Finance, to appear.*
  - Repricing the Cross Smile: An Analytic Joint Density. P.Austing. *preprint 2011*
- **Energy and Commodities**
  - Efficient Pricing of Commodity Options with Early-Exercise under the Ornstein–Uhlenbeck process. C.W.Oosterlee. B.Zhang. *preprint 2011*
  - A finite dimensional approximation for pricing moving average options. M. Bernhart P.Tankov X. Warin, *to appear in SIAM Journal on Financial Mathematics.*
  - Pricing and hedging spread options. R. Carmona V.Durrleman. *SIAM Rev. 45 (2003), no. 4, 627—685.*
  - Closed form spread option valuation. P.Bjerksund G. Stensland *Quantitative Finance, 2011.*
  - A Fourier transform method for spread option pricing. T. R. Hurd and Z. Zhou. *SIAM Journal on Financial Mathematics. 1, 142-–157, 2010.*
  - Multi-asset spread option pricing and hedging *Quantitative Finance, Vol.10, No3, 305-324, 2010.*
  - Unspanned Stochastic Volatility and the Pricing of Commodity Derivatives. A.B.Trolle E.Schwartz. *Rev. Financ. Stud. (2009) 22(11): 4423-4461*
  - Pricing Commodity Swaptions in Multifactor Models. K.Larsson. *The Journal of Derivatives Winter 2011, Vol. 19, No. 2,32-44.*
- **Equity Derivatives**
  - Importance sampling and Statistical Romberg Method. M.B. Alaya A.Kebaier K.Hajji
  - New approximations in local volatility models. E. Gobet, A.Suleiman
  - Componentwise splitting methods for pricing American options under stochastic volatility. Ikonen, S.; Toivanen,J. *Int. J. Theor. Appl. Finance 10 (2007), no. 2, 331–361.*
  - ADI finite difference schemes for option pricing in the Heston model with correlation. K.J. in 't Hout and S. Foulon. *Int. J. Numer. Anal. Mod. 7, 303–320 (2010).*

- ADI schemes with Ikonen-Toivanen splitting for pricing American put options in the Heston model. T. Haentjens, K. in 't Hout and K. Volders. In: Numerical Analysis and Applied Mathematics, eds. T. E. Simos et. al., AIP Conf. Proc. 1281, 231-234 (2010).
- Pricing of Timer Options. C. Bernard Z. Cui. *Journal of Computational Finance, to appear*
- Efficient Simulation of the Double Heston Model. P.Gauthier D.Possamai.
- Greedy methods method for basket options. T.Lelievre J.I.Acevedo
- Pricing higher-dimensional American options using the stochastic grid method. C.W.Oosterlee S. Jain
- Calibration in the Heston model. L. Abbas Turki

The software Premia 15 has been registered at the APP (Agence pour la Protection des Programmes) with the reference IDDN.FR.001.190010.012.S.C.2001.000.31000.

MICMAC Project-Team (section vide)

MOKAPLAN Exploratory Action

5. Software and Platforms

5.1. CFD based MK solvers

5.1.1. Platforms

The core of the ALG2 algorithm [43] for the CFD formulation of the Optimal Mass Transportation problem and many of its generalization is a Poisson solver. Then each problem calls for different but simple modifications the point wise minimization of a given Lagrangian function. We have written such a FreeFem ALG2 platform and are plan to implement a parallel version on Rocquencourt Inria cluster.

5.2. MA based Optimal Mass Transportation solvers

5.2.1. Platforms

Monotone discretisations of the Monge-Ampère operator (the determinant of a Hessian function) is the core of Monge-Ampère Optimal Mass Transportation solvers but is also a useful tool for convexity constraints in infinite dimensional optimization and JKO gradient flows. We are implementing in F90 and comparing several monotone schemes. These modules could be reused in different applications.

SIERRA Project-Team

5. Software and Platforms

5.1. SPAMS (SPArse Modeling Software)

Participants: Jean-Paul Chieze [correspondent], Guillaume Obozinski [correspondent].

SPAMS (SPArse Modeling Software) is an optimization toolbox for solving various sparse estimation problems: dictionary learning and matrix factorization, solving sparse decomposition prob- lems, solving structured sparse decomposition problems. It is developped by Julien Mairal (former Willow PhD student, co-advised by F. Bach and J. Ponce), with the collaboration of Francis Bach (Inria), Jean Ponce (Ecole Normale Supérieure), Guillermo Sapiro (University of Minnesota), Rodolphe Jenatton (Inria) and Guillaume Obozinski (Inria). It is coded in C++ with a Matlab interface. This year, interfaces for R and Python have been developed by Jean-Paul Chieze (engineer Inria). Currently 650 downloads and between 1500 and 2000 page visits per month. See http://spams-devel.gforge.inria.fr/.

5.2. BCFWstruct

Participants: Simon Lacoste-Julien [correspondent], Mark Schmidt.

BCFWstruct is a Matlab implementation of the Block-Coordinate Frank-Wolfe solver for Structural SVMs. See the ICML 2013 paper with the same name.

Participants outside of Sierra: Martin Jaggi (Centre de Mathématiques Appliquées, Ecole Polytechnique); Patrick Pletscher (Machine Learning Laboratory, ETH Zurich)

5.3. SAG

Participant: Mark Schmidt [correspondent].

SAG: Minimizing Finite Sums with the Stochastic Average Gradient.

The SAG code contains C implements (via Matlab mex files) of the stochastic average gradient (SAG) method detailed below, as well as several related methods, for the problem of L2-regularized logistic regression with a finite training set.

The specific methods available in the package are: SGD: The stochastic gradient method with (user-supplied) step-sizes, (optional) projection step, and (optional) (weighted-)averaging. ASGD: A variant of the above code that supports less features, but efficiently implements uniform averaging on sparse data sets. PCD: A basic primal coordinate descent method with step sizes set according the (user-supplied) Lipschitz constants. DCA: A dual coordinate ascent method with a numerical high-accuracy line-search. SAG: The stochastic average gradient method with a (user-supplied) constant step size. SAGlineSearch: The stochastic average gradient method with the line-search described in the paper. SAG-LipschitzLS: The stochastic average gradient method with the line-search and adaptive non-uniform sampling strategy described in the paper.

5.4. fMRI

Participant: Fabian Pedregosa [correspondent].

We showed that HRF estimation improves sensitivity of fMRI encoding and decoding models and propose a new approach for the estimation of Hemodynamic Response Functions from fMRI data. This is an implementation of the methods described in the paper.

ANGE Team

5. Software and Platforms

5.1. FRESHKISS

Although the Saint-Venant system is the cornerstone of flow modelling in geosciences, this does not mean that the transfer of the efficient dedicated simulation tools is achieved in the geoscience community.

ANGE collaborates with scientists, laboratories and companies that are interested in scientific advances which makes the valuation and the transfer of results easier.

The development of robust and efficient numerical tools has been a strong point of the activities within the BANG project-team. ANGE aims at pursuing this effort as most publications of the team members contain both modelling and simulation/validation aspects. For the simulation of the free surface Navier-Stokes equations, numerical tools have been developed namely FRESHKISS2D[1] and FRESHKISS3D. These tools are used by several scientists typically in the BIOCORE Inria project-team, at EDF and in public research laboratories.

FRESHKISS3D is a numerical code solving the 3D hydrostatic and incompressible Navier-Stokes equations with variable density. This code was initially dedicated to research activities within the team but we now aim at turning it into a numerical tool being used by non-mathematicians. Indeed, there is a demand in research laboratories and companies to use this tool. A young engineer (R. Hamouda) has been hired (ADT In@lgae funded by Inria) and its assignment is to improve/enrich the code and to make it user-friendly. Notice that FRESHKISS3D is used for teaching (master students in geosciences) at university Denis Diderot Paris 7 and IPGP.

[1] FRESHKISS: FREe Surface Hydrodynamics using KInetic SchemeS

ARAMIS Team

5. Software and Platforms

5.1. SACHA

Participants: Marie Chupin [Correspondant], Ludovic Fillon.

SACHA ("Segmentation Automatisée Compétitive de l'Hippocampe et de l'Amygdale") is a software for the fully automatic segmentation of the hippocampus and the amygdala from MRI 3D T1 brain scans. It has been validated in various populations including healthy controls and patients with Alzheimer's disease, epilepsy and depression. It has been successfully applied to over 3,000 subjects, both controls, from adolescents to elderly subjects, and patients with different types of pathologies. The current stable version is fully automatic and focused on cross-sectional segmentation. The software can be used both as a command-line program or through a graphical user interface (GUI). The core of the program is coded in C++. It has a dependency to the AIMS library (http://www.brainvisa.info) and preprocessing steps rely on processes in Matlab from SPM (http://www.fil.ion.ucl.ac.uk/spm/). The GUI is coded in Python and is based on BrainVISA (http://www.brainvisa.info). The software has been registered at the APP (French agency for software protection).

5.2. WHASA

Participants: Marie Chupin [Correspondant], Ludovic Fillon, Thomas Samaille.

WHASA ("White matter Hyperintensity Automatic Segmentation Algorithm") is a software for the fully automatic segmentation of age-related white matter hyperintensities from MRI FLAIR and 3D T1 brain scans. It has been validated on a population showing a wide range of lesion load, and is being further evaluated on elderly subjects with few clinical abnormalisties and with different acquisition characteristics. The current stable version is fully automatic and focused on cross-sectional segmentation. The software can be used both as a Matlab command-line or through a graphical user interface (GUI). The core of the program is coded in Matlab. It has a dependency to the SPM environment (http://www.fil.ion.ucl.ac.uk/spm/). The GUI is coded in Python and is based on BrainVISA (http://www.brainvisa.info). The software has been registered at the APP (French agency for software protection).

5.3. Deformetrica

Participants: Stanley Durrleman [Correspondant], Alexandre Routier, Pietro Gori.

Deformetrica is a software which estimates diffeomorphic deformations between sets of geometric objects in 2D and 3D. Those deformations are estimated either for the registration of two of such objects sets or for the construction of an atlas from several of such sets (a template model set and deformations mapping the template model to each set). Geometric objects could be grey-level images, surface meshes, polygonal lines or unstructured point sets. The method relies on the metric on currents for the comparison of point sets and the sum of squared differences for the comparison of images.

The software is written in C++ and relies on the ITK and VTK libraries. It is a command-line software.

The release of the software to the scientific community is planned for 2014.

5.4. qualiCATI

Participants: Marie Chupin [Correspondant], Hugo Dary, Nicolas Vibet, Urielle Thoprakarn, Aude Costard, Amadou Tall, Cyril Poupon, Vincent Perlbarg, Mélanie Pélégrini-Issac.

qualiCATI is a software designed for comprehensive quality control of multimodal MRI data acquisition in large multicentre clinical studies. The software is built as a platform receiving several modules, developped by several CATI engineers. The first module is dedicated to acquisition requirement checking and conversion to nifti format. The second module aims at making 3DT1 acquisition quality check more systematic, and relies both on visual inspection and quantitative indices. The third module allows a simultaneous evaluation of the clinical part of the CATI acquisition protocol. The fourth module embeds automatic indices to evaluate resting state fMRI acquisition. The last module, up to now, is dedicated to first preprocessings and quality indices for dMRI. qualiCATI requires training for the visual parts, and is closely linked with a team of clinical research assistants. It has been used to analyse over 3000 subjects from over 10 multi centre research projects initiated before or after the CATI started. Other modules will be added in the future to embed new aspects of the MRI protocol proposed by the CATI. The Aramis team is in charge of the second and third modules and jointly in charge of the first module. The software is centered on a graphical user interface (GUI). The whole program is coded in Python within the pyPTK environment developped by Cyril Poupon (Neurospin). It has dependencies to SPM (http://www.fil.ion.ucl.ac.uk/spm/) and brainVISA environments as well as specific tools for DICOM management.

BANG Project-Team

5. Software and Platforms

5.1. Software and Platforms

5.1.1. Continuation of M3N

A large part of the software currently in use in the project-team was initiated and developed within former projects (Menusin, M3N).

5.1.2. CellSys

Participants: Géraldine Cellière [PhD student], Dirk Drasdo [correspondent], Stefan Höhme, Adrian Friebel [PhD student, University of Leipzig], Tim Johann [Software Engineer, University of Leipzig], Johannes Neitsch [PhD student], Paul Van Liedekerke [Research Engineer].

Based on an earlier submitted software (Hoehme and Drasdo, Bioinformatics, 2010) a modular computer simulation software for image analysis of tissue samples at histological scales, as well as for individual cell (agent)-based modeling of tumour and tissue growth, and tissue regeneration has been developed. Cell movement is solved either by systems of coupled equations of motion for each individual cell or by Kinetic Monte Carlo methods. The software uses a git framework to facilitate coordinated contributions of multiple developers. The image analysis part allows analysis of structures down to sub-cellular scale such as liver micro-capillaries and bile cannaliculi structures. So far, blood flow as well as growth and regeneration processes, fluxes of chemicals by diffusion and flow etc can be modelled, finite element solvers, ITK and VTK have been integrated.
The software CellSys is calibrated to allow use by external and internal researchers. The idea is to perspectively go open-source and offer consultancy for potential users.
Moreover in 2013 the image processing and analysis chain was refined to capture high resolution laser scanning micrographs. The algorithms were integrated into CELLSYS (see: software) and our experimental partner labs within the projects VLN and NOTOX were provided with the software to allow image analysis directly in their lab and with their people. Along the same line an experimental partner lab at the German Cancer center was provide with a small image analysis tool permitting them to efficiently analyze their bright field images on growing and invasive cancer cell populations in vitro (LUNGSYS).

CLIME Project-Team

5. Software and Platforms

5.1. Polyphemus

Participants: Sylvain Doré, Vivien Mallet, Florian Couvidat [CEREA], Yiguo Wang [CEREA], Nora Duhanyan [CEREA], Yelva Roustan [CEREA].

Polyphemus (see the web site http://cerea.enpc.fr/polyphemus/) is a modeling system for air quality. As such, it is designed to yield up-to-date simulations in a reliable framework: data assimilation, ensemble forecast and daily forecasts. Its completeness makes it suitable for use in many applications: photochemistry, aerosols, radionuclides, etc. It is able to handle simulations from local to continental scales, with several physical models. It is divided into three main parts:

- libraries that gather data processing tools (SeldonData), physical parameterizations (AtmoData) and postprocessing abilities (AtmoPy);
- programs for physical preprocessing and chemistry-transport models (Polair3D, Castor, two Gaussian models, a Lagrangian model);
- model drivers and observation modules for model coupling, ensemble forecasting and data assimilation.

Figure 1 depicts a typical result produced by Polyphemus.

Clime is involved in the overall design of the system and in the development of advanced methods in model coupling, data assimilation and uncertainty quantification (through model drivers and post-processing).

In 2013, Polyphemus has received numerous improvements on aerosol modeling, including better dynamics for organic aerosol formation and interactions between organic and inorganic aerosols. The data assimilation part of Polyphemus can now perform 3D data assimilation, taking advantage of Lidar data. Further integration of the data assimilation library Verdandi was also carried out.

5.2. Data assimilation library: Verdandi

Participants: Vivien Mallet, Dominique Chapelle [M3DISIM], Philippe Moireau [M3DISIM], Anne Tilloy, Paul Baudin, Tristan Perotin.

The leading idea is to develop a data assimilation library intended to be generic, at least for high-dimensional systems. Data assimilation methods, developed and used by several teams at Inria, are generic enough to be coded independently of the system to which they are applied. Therefore these methods can be put together in a library aiming at:

- making easier the application of methods to a great number of problems,
- making the developments perennial and sharing them,
- improving the broadcast of data assimilation works.

An object-oriented language (C++) has been chosen for the core of the library. A high-level interface to Python is automatically built. The design study raised many questions, related to high dimensional scientific computing, the limits of the object contents and their interfaces. The chosen object-oriented design is mainly based on three class hierarchies: the methods, the observation managers and the models. Several base facilities have also been included, for message exchanges between the objects, output saves, logging capabilities, computing with sparse matrices.

In 2013, version 1.5 was released with better consistency between the methods. Verdandi received improvements in its test cases. Increased flexibility was introduced in error descriptions, especially for uncertainty quantification.

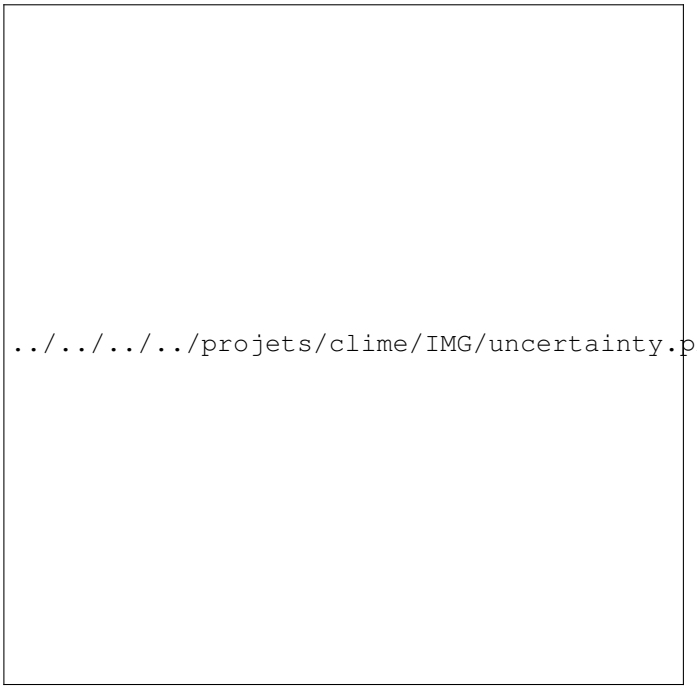

Figure 1. Map of the relative standard deviation (or spread, %) of an ensemble built with Polyphemus (ozone simulations, $\mu g\ m^{-3}$). The standard deviations are averaged over the summer of 2001. They provide an estimation of the simulation uncertainties.

A C++ interface to the Nucleus for European Modelling of the Ocean (see the web site NEMO http://www.nemo-ocean.eu/) has been developed so that it can be plugged to Verdandi. The interface currently enables the application of Monte Carlo simulations and the ensemble Kalman filter.

5.3. Urban air quality analysis

Participants: Anne Tilloy, Vivien Mallet, Raphaël Périllat.

"Urban Air Quality Analysis" carries out data assimilation at urban scale. It merges the outputs of a numerical model (maps of pollutant concentrations) with observations from an air quality monitoring network, in order to produce the so-called analyses, that is, corrected concentration maps. The data assimilation computes the Best Linear Unbiased Estimator (BLUE), with a call to the data assimilation library Verdandi. The error covariance matrices are parameterized for both model simulations and observations. For the model state error covariances, the parameterization primarily relies on the road network. The software handles ADMS Urban output files, for a posteriori analyses or in an operational context.

In 2013, the software introduced new models for error covariances. It may now take into account tunnels. New options were added to filter out certain observations. The software was extended to handle new file formats.

POMDAPI Project-Team

4. Software and Platforms

4.1. LifeV

Participant: Michel Kern.

LifeV is a finite element (FE) library providing implementations of state of the art mathematical and numerical methods. It serves both as a research and production library. It has been used already in medical and industrial context to simulate fluid structure interaction and mass transport. LifeV is the joint collaboration between four institutions: École Polytechnique Fédérale de Lausanne (CMCS) in Switzerland, Politecnico di Milano (MOX) in Italy, Inria (Pomdapi) in France and Emory University (Sc. Comp) in the U.S.A.

Version 3.1.1

Programming language: C++

http://www.lifev.org/

4.2. M1cg1

Participant: Jean Charles Gilbert.

M1cg1 solves convex quadratic optimization problems and builds preconditioning matrices.

Version: 1.2

Programming language: Fortran 77

14 downloads in 2013

https://who.rocq.inria.fr/Jean-Charles.Gilbert/modulopt/optimization-routines/m1cg1/m1cg1.html

4.3. M1qn3

Participant: Jean Charles Gilbert.

M1qn3 solves very large scale differentiable optimization problems.

Version: 3.3

Programming language: Fortran 77

36 downloads in 2013

https://who.rocq.inria.fr/Jean-Charles.Gilbert/modulopt/optimization-routines/m1qn3/m1qn3.html

In collaboration with Claude Lemaréchal (project-team Bipop)

4.4. Oqla, Qpalm

Participants: Jean Charles Gilbert, Émilie Joannopoulos.

Oqla and Qpalm aim at solving large scale convex quadratic functions on a polyhedron by an augmented Lagrangian method.

Versions (in development): 0.1 (Oqla), 0.2 (Qpalm)

Programming languages: `C++` (Oqla), `Matlab` (Qpalm)

4.5. Ref-image

Participants: Hend Ben Ameur, François Clément, Pierre Weis.

Ref-image is an image segmentation program using optimal control techniques. Slogan is “no gestalt inside”. Ref-image implements the refinement indicator algorithm, specialized to the case of the inversion of the identity map. It is a first step towards the implementation of a generic inversion platform using the refinement indicator algorithm.

Version: 1.0+pl0

Programming language: OCaml

http://refinement.inria.fr/ref-image/

4.6. SQPlab

Participant: Jean Charles Gilbert.

SQPlab solves constrained differentiable optimization problems.

Version: 0.4.5

Programming language: Matlab

200 downloads in 2013

https://who.rocq.inria.fr/Jean-Charles.Gilbert/modulopt/optimization-routines/sqplab/sqplab.html

4.7. Sklml

Participants: François Clément, Pierre Weis.

Sklml is a functional parallel skeleton compiler and programming system for OCaml programs. Slogan is “easy coarse grain parallelization”.

Version: 1.1+pl0

Programming language: OCaml

http://sklml.inria.fr/

4.8. FreeFem++

Participants: Martin Vohralík, Martin Čermák, Zuqi Tang.

The scientific calculation code FreeFem++ is an excellent example of a complex software numerical simulation tool. It in particular encompasses all specification of the problem, the choice and implementation of the numerical method, the choice and implementation of the linearization method (nonlinear solver), and the choice and implementation of the method of solution of the associated linear systems (linear solver). In the post-doc stays of M. Čermák and Z. Tang, we integrate there the most recent advances of the theory of a posteriori error estimation and of adaptive algorithms. In particular, adaptive stopping criteria for the linear and nonlinear solvers are being implemented.

Version 3.26-2

Programming language: C++

http://www.freefem.org/ff++/

REO Project-Team

5. Software and Platforms

5.1. LiFE-V library

Participants: Miguel Ángel Fernández Varela [correspondant], Jean-Frédéric Gerbeau.

LiFE-V [2] is a finite element library providing implementations of state of the art mathematical and numerical methods. It serves both as a research and production library. LiFE-V is the joint collaboration between three institutions: Ecole Polytechnique Fédérale de Lausanne (CMCS) in Switzerland, Politecnico di Milano (MOX) in Italy and Inria (REO) in France. It is a free software under LGPL license.

5.2. Mistral library

Participant: Jean-Frédéric Gerbeau [correspondant].

Mistral is a finite element library which implements in particular fluid-structure interaction algorithms (ALE and Fictitious domain formulations), fluid surface flow (ALE) and incompressible magnetohydrodynamics equations. Mistral results from a collaboration between Inria and ENPC (CERMICS).

5.3. FELiScE

Participants: Grégory Arbia, Cesare Corrado, Miguel Ángel Fernández Varela, Justine Fouchet-Incaux, David Froger, Jean-Frédéric Gerbeau [correspondant], Damiano Lombardi, Elisa Schenone, Saverio Smaldone, Marina Vidrascu, Irène Vignon-Clementel.

FELiScE – standing for "Finite Elements for Life Sciences and Engineering" – is a new finite element code which the MACS and REO project-teams have decided to jointly develop in order to build up on their respective experiences concerning finite element simulations. One specific objective of this code is to provide in a unified software environment all the state-of-the-art tools needed to perform simulations of the complex cardiovascular models considered in the two teams – namely involving fluid and solid mechanics, electrophysiology, and the various associated coupling phenomena. FELISCE is written in C++, and may be later released as an opensource library. https://gforge.inria.fr/projects/felisce/

5.4. SHELDDON

Participant: Marina Vidrascu [correspondant].

SHELDDON (SHELls and structural Dynamics with DOmain decomposition in Nonlinear analysis) is a finite element library based on the Modulef package which contains shell elements, nonlinear procedures and PVM subroutines used in domain decomposition or coupling methods, in particular fluid-structure interaction. (https://gforge.inria.fr/projects/shelddon)

[2] http://www.lifev.org/

SISYPHE Project-Team

5. Software and Platforms

5.1. SITB: The Matlab System Identification ToolBox

Participant: Qinghua Zhang.

This development is made in collaboration with Lennart Ljung (Linköping University, Sweden), Anatoli Juditsky (Joseph Fourier University, France) and Peter Lindskog (NIRA Dynamics, Sweden).

The System Identification ToolBox (SITB) is one of the main Matlab toolboxes commercialized by The Mathworks. Inria participates in the development of its extension to the identification of nonlinear systems which is released since 2007. It includes algorithms for both black box and grey box identification of nonlinear dynamic systems. Inria is mainly responsible for the development of black box identification, with nonlinear autoregressive (NLARX) models and block-oriented (Hammerstein-Wiener) models.

5.2. ISTL: Inverse Scattering for Transmission Lines

Participants: Michel Sorine, Qinghua Zhang.

ISTL is a software for numerical computation of the inverse scattering transform for electrical transmission lines. In addition to the inverse scattering transform, it includes a numerical simulator generating the reflection coefficients of user-specified transmission lines. With the aid of a graphical interface, the user can interactively define the distributed characteristics of a transmission line. It is registered at Agence pour la Protection des Programmes (APP) under the number IDDN.FR.001.120003.000.S.P.2010.000.30705.

5.3. CGAO: Contrôle Glycémique Assisté par Ordinateur

Participants: Alexandre Guerrini, Michel Sorine.

The software CGAO developed with LK2 and P. Kalfon (Hospital Louis Pasteur, Chartres) provides efficient monitoring and control tools that will help physicians and nursing staff to avoid hyperglycaemia and hypoglycaemia episodes in Intensive Care Units. The controller determines the insulin infusion rate, glucose bolus and scheduling of blood glucose measurement on the basis of the standard available glycaemia measurements. A first version, CGAO_v1, has been used in a large clinical study CGAO-REA (see Section 6.3.1). An improved version, CGAO_v2 registered at APP under the number IDDN.FR.001.360019.002.S.P.2009.000.31230 is now used by the company Fresenius Kabi (see Section 7.1).

5.4. DYNPEAK: a Scilab toolbox and a Web service for the analysis of LH (Luteinizing Hormone) secretion rhythms

Participants: Frédérique Clément, Claire Médigue, Serge Steer, Mouhamadoul Bachir Syll, Alexandre Vidal, Qinghua Zhang.

DYNPEAK is a software dedicated to the analysis of the pulsatile rhythm of secretion of the pituitary hormone LH, that aims at providing the final users (experimentalists and clinicians) with a simple-to-use version of the algorithm developed in [25]. It has been implemented as a Scilab atom toolbox (http://atoms.scilab.org/toolboxes/Dynpeak) and registered in APP under the reference DynPeak V1.0, IDDN.FR.001.360015.000.S.P.2013.000.10000. The web service version of DynPeak (https://dynpeak.inria.fr), still in test, has also been updated and a new release is planned for a next future.

5.5. The Cardiovascular toolbox for Scilab

Participants: Claire Médigue, Michel Sorine, Serge Steer.

This Cardiovascular toolbox is an “atom” of Scilab developed by Serge Steer to distribute the cardiovascular signal processing tools designed and intensively used internally in the team for several years by Claire Médigue, Alessandro Monti and Michel Sorine. It includes baroreflex analysis using a multi channel non stationary signal analysis method ; the cardiovascular signal spectral analysis using time-frequency decomposition and signal demodulations methods, e.g. for respiratory sinus arrhythmia analysis. It replaces LARY_CR, the former software package dedicated to the study of cardiovascular and respiratory rhythms [108].

5.6. K-Assessor: assessment of controllers

Participants: Habib Jreige, Michel Sorine.

This development is made in collaboration with the small business enterprise SciWorks Technologies (Jim Pioche). We have defined a method to assess SISO (Single Input / Single Output) controllers based on square or cubic tables of metadata easily manipulated on a computer and easily interpretable by control experts and field experts (emergency doctors in our case) who can use them to jointly tune a risk estimator. The agreement between experts is obtained using a ROC-analysis approach. The software K-Assessor implements this methodology. It is registered at APP under the number IDDN.FR.001.390011.000.S.P.2013.000.10000.

ALPINES Team

5. Software and Platforms

5.1. Software and Platforms

5.1.1. Platforms

5.1.1.1. FreeFem++, http://www.freefem.org/ff++/

`FreeFem++` is a PDE solver based on a flexible language that allows a large number of problems to be expressed (elasticity, fluids, etc) with different finite element approximations on different meshes. There are more than 2000 users, and on the mailing list there are 430 members. Among those, we are aware of at least 10 industrial companies, 8 french companies and 2 non-french companies. It is used for teaching at Ecole Polytechnique, Ecole Centrale, Ecole des Ponts, Ecole des Mines, University Paris 11, University Paris Dauphine, La Rochelle, Nancy, Metz, Lyon, etc. Outside France, it is used for example at universities in Japan (Tokyo, Kyoto, Hiroshima, there is a userguide FreeFem++ in japan), Spain (Sevilla, BCAM, userguide available in spanish), UK (Oxford), Slovenia, Switzerland (EPFL, ETH), China. For every new version, there are 350 regression tests, and we provide a rapid correction of reported bugs. The licence of FreeFem++ is LGPL.

5.1.1.2. Library for preconditioned iterative methods

In the project-team we develop a library that integrates the direction preserving and low rank approximation preconditioners for both approached factorizations and domain decomposition like methods. It will be available through `FreeFem++` and also as a stand alone library, and we expect to have one version of this library available in 2014.

ARLES Project-Team

5. Software and Platforms

5.1. Introduction

In order to validate our research results, our research activities encompass the development of related prototypes as surveyed below.

5.2. iCONNECT – Emergent Middleware Enablers

Participant: Valérie Issarny [correspondent].

As part of our research work on Emergent Middleware, we have implemented Enablers (or Enabler functionalities) that make part of the overall CONNECT architecture realizing Emergent Middleware in practice [2]. The focus of ARLES work is on the: *Discovery enabler* that builds on our extensive background in the area of interoperable pervasive service discovery; and *Synthesis enabler* that synthesizes mediators that allow Networked Systems (NSs) that have compatible functionalities to interact despite mismatching interfaces and/or behaviors.

The Discovery Enabler is the component of the overall CONNECT architecture that handles discovery of networked systems (NSs), stores their descriptions (NS models), and performs an initial phase of matchmaking to determine which pairs of systems are likely to be able to interoperate. Such pairs are then passed to the Synthesis Enabler so that mediators can be generated. The Discovery Enabler is written in Java and implements several legacy discovery protocols including DPWS and UPnP.

The Synthesis Enabler assumes semantically-annotated system descriptions *à la* OWL-S, which are made available by the Discovery Enabler, together with a domain ontology, and produces the mediators that enable functionally compatible networked systems to interoperate. The semantically-annotated interfaces of the NSs that need to communicate are processed to compute the semantic mapping between their respective operations using a constraint solver. The resulting mapping serves generating a mediator that coordinates the behaviors of the NSs and guarantees their successful interaction. Only when the mediator includes all the details about the communication of NSs, can interoperability be achieved, which calls for the adequate concretization of synthesized mediators.

The *concretization of mediators* bridges the gap between the application level, which provides the abstraction necessary to reason about interoperability and synthesize mediators, and the middleware-level, which provides the techniques necessary to implement these mediators. Concretization entails the instantiation of the data structures expected by each NS and their delivery according to the interaction pattern defined by the middleware, based on which the NS is implemented. Therefore, we have been developing a mediation engine that, besides executing the data translations specified by the mediator, generates composed parsers and composers, which can process complex messages, by relying on existing libraries associated with standard protocols and state-of-the-art middleware solutions.

The Discovery and Synthesis Enablers have been integrated and experimented with by the CONNECT consortium to effectively enable Emergent Middleware. Part of them are available for download under an open source license at the CONNECT Web site at https://www.connect-forever.eu/software.html.

5.3. Service-oriented Middleware for Pervasive Computing

Participants: Nikolaos Georgantas [correspondent], Valérie Issarny [correspondent].

In the past years, we have built a strong foundation of service-oriented middleware to support the pervasive computing vision. This specifically takes the form of a family of middlewares, all of which have been released under the open source LGPL license:

- **WSAMI - A Middleware Based on Web Services for Ambient Intelligence:** WSAMI (Web Services for AMbient Intelligence) is based on the Web services architecture and allows for the deployment of services on wireless handheld devices like smartphones.
  URL: http://www-rocq.inria.fr/arles/download/ozone/index.htm
- **Ariadne - A Protocol for Scalable Service Discovery in MANETs:** Ariadne enriches WSAMI with the Ariadne service discovery protocol, which has been designed to support decentralized Web service discovery in multi-hop mobile ad hoc networks (MANETs). Ariadne enables small and resource-constrained mobile devices to seek and find complementary, possibly mobile, Web services needed to complete specified tasks in MANETs, while minimizing the traffic generated and tolerating intermittent connectivity.
  URL: http://www-rocq.inria.fr/arles/download/ariadne/index.html
- **MUSDAC - A Middleware for Service Discovery and Access in Pervasive Networks:** The MUlti-protocol Service Discovery and ACcess (MUSDAC) middleware platform enriches WSAMI so as to enable the discovery and access to services in the pervasive environment, which is viewed as a loose and dynamic composition of independent networks. MUSDAC manages the efficient dissemination of discovery requests between the different networks and relies on specific plug-ins to interact with the various middleware used by the networked services.
  URL: http://www-rocq.inria.fr/arles/download/ubisec/index.html
- **INMIDIO - An Interoperable Middleware for Ambient Intelligence:** INMIDIO (INteroperable MIddleware for service Discovery and service InteractiOn) dynamically resolves middleware mismatch. More particularly, INMIDIO identifies the interaction middleware and also the discovery protocols that execute on the network and translates the incoming/outgoing messages of one protocol into messages of another, target protocol.
  URL: http://www-rocq.inria.fr/arles/download/inmidio/index.html
- **COCOA - A Semantic Service Middleware:** COCOA is a comprehensive approach to semantic service description, discovery, composition, adaptation and execution, which enables the integration of heterogeneous services of the pervasive environment into complex user tasks based on their abstract specification. Using COCOA, abstract user tasks are realized by dynamically composing the capabilities of services that are currently available in the environment.
  URL: http://gforge.inria.fr/projects/amigo/
- **ubiSOAP - A Service Oriented Middleware for Seamless Networking:** ubiSOAP brings multi-radio, multi-network connectivity to services through a comprehensive layered architecture: (i) the multi-radio device management and networking layers together abstract multi-radio connectivity, selecting the optimal communication link to/from nodes, according to quality parameters; (ii) the communication layer allows for SOAP-based point-to-point and group-based interactions in the pervasive network; and (iii) the middleware services layer brings advanced distributed resource management functionalities customized for the pervasive networking environment.
  URL: http://www.ist-plastic.org.

5.4. XSB – eXtensible Service Bus for the Future Internet

Participant: Nikolaos Georgantas [correspondent].

The eXtensible Service Bus (XSB) is a development and runtime environment dedicated to complex distributed applications of the Future Internet. Such applications will be based, to a large extent, on the open integration of extremely heterogeneous systems, such as lightweight embedded systems (e.g., sensors, actuators and networks of them), mobile systems (e.g., smartphone applications), and resource-rich IT systems (e.g., systems hosted on enterprise servers and Cloud infrastructures). Such heterogeneous systems are supported by enabling middleware platforms, particularly for their interaction. With regard to middleware-supported interaction, the client-service (CS), publish-subscribe (PS), and tuple space (TS) paradigms are among the most widely employed ones, with numerous related middleware platforms, such as: Web Services, Java RMI for CS; JMS, SIENA for PS; and JavaSpaces, Lime for TS. XSB then provides support for the seamless integration of heterogeneous interaction paradigms (CS, PS and TS).

In a nutshell, our systematic interoperability approach implemented by the proposed XSB is carried out in two stages. First, a middleware platform is abstracted under a corresponding interaction paradigm among the three base ones, i.e., CS, PS and TS. To this aim, we have elicited a connector model for each paradigm, which comprehensively covers its essential semantics. Then, these three models are abstracted further into a single generic application (GA) connector model, which encompasses their common interaction semantics. Based on GA, we build abstract connector converters that enable interconnecting the base interaction paradigms.

Following the above, XSB is an abstract service bus that prescribes only the high-level semantics of the common bus protocol, which is the GA semantics. Furthermore, we provide an implementation of the XSB, building upon existing SOA and ESB realizations. XSB features richer interaction semantics than common ESBs to deal effectively with the increased Future Internet heterogeneity. Moreover, from its very conception, XSB incorporates special consideration for the cross-integration of heterogeneous interaction paradigms. Services relying on different interaction paradigms can be plugged into XSB by employing binding components (BCs) that adapt between their native middleware and the common bus protocol. This adaptation is based on the abstractions, and in particular on the conversion between the native middleware, the corresponding CS/PS/TS abstraction, and the GA abstraction.

Furthermore, we provide a companion implementation, named Light Service Bus (LSB), targeting the Internet of Things(IoT) domain. LSB forms a concrete access solution for IoT systems as it is able to cope with the diversity of the involved interaction protocols and take care of the IoTS specifics, such as resource constraints, dynamic environments, data orientation, etc. It is implemented to be lightweight in nature and uses REST as the common protocol/bus in place of an ESB solution. In LSB, we confirm the wide use of the aforementioned interaction paradigms (CS/PS/TS) but also underline the existence of an additional paradigm focused on continuous interaction known as Streaming (STR).

Both the XSB and LSB solutions are available for download under open source licenses at http://xsb.inria.fr and http://websvn.ow2.org/listing.php?repname=choreos&path=%2Ftrunk%2Fextensible-service-access%2Flsb%2FLsbBindingComponents%2F respectively.

5.5. MobIoT – Service-oriented Middleware for the Mobile IoT

Participant: Valérie Issarny [correspondent].

MobIoT is a service-oriented middleware aimed at the mobile Internet of Things (IoT), which in particular deals with the ultra-large scale, heterogeneity and dynamics of the target networking environment. MobIoT offers novel probabilistic service discovery and composition approaches, and wraps legacy access protocols to be seamlessly executed by the middleware. The middleware exposes two levels of service abstractions: Thing as a service (on the service provider side); and Things measurements/actions as a service (on the service consumer side).

Key features of MobIoT lie in: (i) the exploitation of ontologies to overcome the heterogeneity of the Things network, (ii) the introduction of probabilistic approaches for both registering and retrieving networked things so as to filter out the ones that are redundant with already known alternatives, and finally, (iii) the exploitation of Thing services composition for responding to user queries asking information about the physical world so as to ease interaction with such a complex and dynamic networking environment.

MobIoT is implemented using Java and the Android platform, and consists of two complementary components: The MobIoT Mobile middleware and the MobIoT Web Service. The MobIoT Mobile middleware is deployed on mobile devices (e.g., smartphones, tablets, sensor devices). It wraps: (i) the Query component that enables the querying of the physical world, (ii) the Registration component that deals with the probabilistic registration of local sensors and actuators, (iii) the domain ontology that allows reasoning about the features of Things, and (iv) the Sensor Access component that enables the sensor data retrieval and exposure. The MobIoT Web Service wraps: (i) the Registry component that keeps tracks of the registered services, (ii) the probabilistic Lookup component that allow retrieving relevant services in a scalable way, and (iii) the Composition & Estimation component to answer queries over the physical world using available Thing services, and finally domain and devices ontologies.

The MobIoT middleware is available for download under an open source license at http://choreos.eu/bin/Documentation/IoTS_Middleware.

5.6. Srijan: Data-driven Macroprogramming for Sensor Networks

Participant: Animesh Pathak [correspondent].

Macroprogramming is an application development technique for wireless sensor networks (WSNs) where the developer specifies the behavior of the system, as opposed to that of the constituent nodes. As part of our work in this domain, we are working on *Srijan*, a toolkit that enables application development for WSNs in a graphical manner using data-driven macroprogramming.

It can be used in various stages of application development, *viz.,*

1. Specification of application as a task graph,
2. Customization of the auto-generated source files with domain-specific imperative code,
3. Specification of the target system structure,
4. Compilation of the macroprogram into individual customized runtimes for each constituent node of the target system, and finally
5. Deployment of the auto generated node-level code in an over-the-air manner to the nodes in the target system.

The current implementation of *Srijan* targets both the Sun SPOT sensor nodes and larger nodes with J2SE. Most recently, *Srijan* also includes rudimentary support for incorporating Web services in the application being designed.

The software is released under open source license, and available as an Eclipse plug-in at http://code.google.com/p/srijan-toolkit/.

5.7. Yarta: Middleware for supporting Mobile Social Applications

Participant: Animesh Pathak [correspondent].

With the increased prevalence of advanced mobile devices (the so-called “smart” phones), interest has grown in *Mobile Social Ecosystems* (MSEs), where users not only access traditional Web-based social networks using their mobile devices, but are also able to use the context information provided by these devices to further enrich their interactions. We are developing a middleware framework for managing mobile social ecosystems, having a multi-layer middleware architecture consisting of modules, which will provide the needed functionalities, including:

- Extraction of social ties from context (both physical and virtual),
- Enforcement of access control to protect social data from arbitrary access,
- A rich set of MSE management functionalities, using which mobile social applications can be developed.

Our middleware adopts a graph-based model for representing social data, where nodes and arcs describe socially relevant entities and their connections. In particular, we exploit the Resource Description Framework (RDF), a basic Semantic Web standard language that allows representing and reasoning about social vocabulary, and creating an interconnected graph of socially relevant information from different sources.

The current implementation of the Yarta middleware targets both desktop/laptop nodes running Java 2 SE, as well as Android smart phones.

The software is released under open source license at https://gforge.inria.fr/projects/yarta/.

5.8. iBICOOP: Mobile Data Management in Multi-* Networks

Participant: Valérie Issarny [correspondent].

Building on the lessons learned with the development of pervasive service oriented middleware and of applications using them, we have been developing the custom iBICOOP middleware. iBICOOP specifically aims at assisting the development of advanced mobile, collaborative application services by supporting interactions between mobile users.

Briefly, the iBICOOP middleware addresses the challenges of easily accessing content stored on mobile devices, and consistent data access across multiple mobile devices by targeting both fixed and mobile devices, leveraging their characteristics (e.g., always on and unlimited storage for home/enterprise servers, ad hoc communication link between mobile devices), and by leveraging the capabilities of all available networks (e.g., ad hoc networks, Internet, Telecoms infrastructure networks). It also relies on Web and Telecoms standards to promote interoperability.

The base architecture of the iBICOOP middleware consists of core modules on top of which we can develop applications that may arise in the up-coming multi-device, multi-user world:

- The *Communication Manager* provides mechanisms to communicate over different available network interfaces of a device — Bluetooth, WiFi, Cellular — and also using different technologies e.g., Web services, HTTP/TCP sockets, ad hoc mode.
- The *Security Manager* uses well-established techniques of cryptography and secure communication to provide necessary security.
- The *Partnership Manager* provides device or user information in the form of *profiles*.
- iBICOOP relies on service location protocols for *naming and discovery* of nearby services on currently active network interfaces that support IP multicast.
- Besides normal file managing tasks, the *Local File Manager* gives the user clear cues to the files that have been replicated across multiple devices or shared among different users by using different icons.

The iBICOOP middleware has been licensed by AMBIENTIC (http://www.ambientic.com/), a start-up that specifically develops innovative mobile distributed services on top of the iBICOOP middleware that allows for seamless interaction and content sharing in today's multi-* networks.

DYOGENE Project-Team

5. Software and Platforms

5.1. SINR-based k-coverage probability in cellular networks

H. P. Keeler, "SINR-based k-coverage probability in cellular networks" MATLAB Central File Exchange, 2013.

Available at: http://www.mathworks.fr/matlabcentral/fileexchange/40087-sinr-based-k-coverage-probability-in-cellular-networks

The scripts calculate the SINR k-coverage probability in a single-tier cellular network using a method based on a homogeneous Poisson process model. Details can be found in [20] which presents the model that these scripts are based on.

GANG Project-Team (section vide)

HIPERCOM2 Team

5. Software and Platforms

5.1. Software and Platforms

5.1.1. Platforms

5.1.1.1. SensLab and FIT

Participants: Cédric Adjih, Alaeddine Weslati, Vincent Ladeveze.

This is a joint work with Emmanuel Baccelli from Inria Saclay.

Period: 2011 - 2021

Partners: Inria (Lille, Sophia-Antipolis, Grenoble), INSA, UPMC, Institut Télécom Paris, Institut Télécom Evry, LSIIT Strasbourg.

- Deployment: during the year 2013, most of the practical deployment has been planned, designed and realized. A location has been found for the new testbed of the EQUIPEX FIT: the basement of building 1 at Rocquencourt. The preparation of the deployment space, power, GPS and network infrastructures for the deployment of IoT-Lab testbed has been finished: including designing and installing support structures, amenaging space (ventilation, ...), acquiring and installing network equipment, servers, GPS, ...
  The Senslab testbed has been moved to the building 1 and has been integrated into the new platform. Deployment is in a finalization phase and should go in beta testing starting from Q1 2014. The testbed will offer 344 open nodes, including 120 WSN430 nodes, 200 Cortex A8 based nodes, 24 Cortex M3.
- Support of external projects: Support for RIOT-OS and OpenWSN projects has been developed for IoT-Lab hardware and is being tested.
  - RIOT-OS, a joint work between Inria and FU-Berlin to create an Operating System for the Internet of Things.
  - OpenWSN, an open-source protocol stack for Internet of Things developed by UC Berkley.
  - IoT-Lab hardware based on STM stm32f1 series ARM Cortex-M3 MCU and Atmel AT86RF231 radio transceiver.

5.1.2. Software

5.1.2.1. NS3

Participants: Cédric Adjih, Hana Baccouch.

Ey-Wifi, Elimination-Yield for WiFi networks, is a module developed for the ns-3 simulation tool to integrate the features of the EY-NPMA channel access scheme. EY-NPMA (Elimination-Yield Non-Pre-emptive Priority Multiple Access) is a contention based protocol using active signaling (black burst): a node requests access to the medium by transmitting a burst signal. More precisely, the channel access cycle comprises three phases: priority phase, elimination phase and yield phase.

This software was developped thanks to the ADT MOBSIM.

The Ey-Wifi module has been publically released and is available, along with a detailed tutorial explaining how to use it, at: http://hipercom.inria.fr/Ey-Wifi

5.1.2.2. OPERA

Participants: Cédric Adjih, Ichrak Amdouni, Pascale Minet, Saoucene Ridene, Ridha Soua.

The OPERA software was developed by the Hipercom2 team in the OCARI project. They include EOLSR, an energy efficient routing protocol and OSERENA, a coloring algorithm optimized for dense wireless networks. OPERA was registered by the APP. In 2013, OPERA has been made available for download as an open software from the InriaGForge site: https://gforge.inria.fr/scm/?group_id=4665

More details and documentation about this software are available in the website made by the Hipercom2 team: http://opera.gforge.inria.fr/index.html

5.1.2.3. SAHARA

Participants: Erwan Livolant, Pascale Minet, Ridha Soua, Cédric Adjih.

The software modules developed by the Hipercom2 team in the SAHARA project have been registered by the APP in July 2013:

- Mundi-Safeti V1.0, Reference: IDDN.FR.001.270022.000.S.P.2013.000.10000
- SAHARA-Network V1.0, Reference: IDDN.FR.001.270021.000.S.P.2013.000.10000

RAP Project-Team (section vide)

REGAL Project-Team

4. Software and Platforms

4.1. Coccinelle

Participants: Christian Clausen, Julia Lawall [correspondent], Gilles Muller [correspondent], Suman Saha, Gaël Thomas.

Coccinelle is a program matching and transformation engine which provides the language SmPL (Semantic Patch Language) for specifying desired matches and transformations in C code. Coccinelle was initially targeted towards performing collateral evolutions in Linux. Such evolutions comprise the changes that are needed in client code in response to evolutions in library APIs, and may include modifications such as renaming a function, adding a function argument whose value is somehow context-dependent, and reorganizing a data structure.

Beyond collateral evolutions, Coccinelle has been successfully used for finding and fixing bugs in systems code. One of the main recent results is an extensive study of bugs in Linux 2.6 that has permitted us to demonstrate that the quality of code has been improving over the last six years, even though the code size has more than doubled.

Coccinelle is freely available at http://coccinelle.lip6.fr under a GPL v2 license.

4.2. SwiftCloud

Participants: Mahsa Najafzadeh, Burcu Külahçioglu Özkan, Marc Shapiro [correspondent], Marek Zawirski.

Cloud computing infrastructures improve latency and provide high availability by geo-replicating data at different locations across the world. This improves user experience, which is important for services such as social networks, online shops and games. Nevertheless, the distance to the closest data centre is still far from optimal for many users.

SwiftCloud is the first system to bring geo-replication all the way to the client machine, in order to provide the best possible latency and availability. This raises two main challenges. One, how to provide, efficiently, convenient programming guarantees, including access to consistent data in read-write transactions, and ensuring session guarantees. Two, to continue providing these guarantees, despite failures that require a client to switch to a different data centre.

Our research report [61] presents the design of SwiftCloud and the algorithms it uses to achieve the desired properties, while aiming for efficiency and for scalability to large numbers of clients. Our evaluation confirms that its programming model is practical, and that its performance and fault tolerance objectives are met.

SwiftCloud is supported by the ConcoRDanT ANR project (Section 7.1.6), by a Google European Doctoral Fellowship, and by the new FP7 grant SyncFree (Section 7.2.1.1).

The code is freely available on http://gforge.inria.fr/ under a BSD license.

4.3. JESSY

Participants: Masoud Saeida Ardekani, Marc Shapiro [correspondent].

A large family of distributed transactional protocols have a common structure, called Deferred Update Replication (DUR). DUR provides dependability by replicating data, and performance by not re-executing transactions but only applying their updates. Protocols of the DUR family differ only in behaviors of few generic functions. Based on this insight, we offer a generic DUR framework, called Jessy, along with a library of finely-optimized plug-in implementations of the required behaviors. Our empirical study shows that:

1. The framework provides a fair, apples-to-apples comparison between transactional protocols;
2. By replacing plugs-ins, developers can use Jessy to understand bottlenecks in their protocols;
3. This in turn enables the improvement of existing protocols; and
4. Given a protocol, Jessy allows to evaluate the cost of ensuring various degrees of dependability.

Articles related to Jessy were published in an Inria research report [60], in the Symp. on Reliable Distr. Sys. (SRDS) [43] and in the Euro-Par conference [42] Jessy is supported by a UPMC PhD scholarship to Masoud Saeida Ardekani, and by the ConcoRDanT ANR project (Section 7.1.6).

Jessy is freely available on github under http://Github.com/msaeida/jessy under an Apache license.

4.4. Java and .Net runtimes for LLVM

Participants: Koutheir Attouchi, Harris Bakiras, Bertil Folliot, Julia Lawall, Gilles Muller, Thomas Preud'Homme, Gaël Thomas [correspondent].

Many systems research projects now target managed runtime environments (MREs) because they provide better productivity and safety compared to native environments. Still, developing and optimizing an MRE is a tedious task that requires many years of development. Although MREs share some common functionalities, such as a Just In Time Compiler or a Garbage Collector, this opportunity for sharing implementations has not been yet exploited in implementing MREs. We are working on VMKit, a first attempt to build a common substrate that eases the development and experimentation of high-level MREs and systems mechanisms.

VMKit has been used to implement a JVM and a CLI Virtual Machine (Microsoft .NET is an implementation of the CLI) using the LLVM compiler framework and the MMTk garbage collectors. The JVM, called J3, executes real-world applications such as Tomcat, Felix or Eclipse and the DaCapo benchmark. It uses the GNU Classpath project for the base classes. The CLI implementation, called N3, is its in early stages but can execute simple applications and the "pnetmark" benchmark. It uses the pnetlib project or Mono as its core library. The VMKit VMs compare in performance with industrial and top open-source VMs on CPU-intensive applications. VMKit is publicly available under the LLVM license.

http://vmkit2.gforge.inria.fr/

ALPAGE Project-Team

5. Software and Platforms

5.1. Syntax

Participants: Pierre Boullier [correspondant], Benoît Sagot.

See also the web page http://syntax.gforge.inria.fr/.

The (currently beta) version 6.0 of the SYNTAX system (freely available on Inria GForge) includes various deterministic and non-deterministic CFG parser generators. It includes in particular an efficient implementation of the Earley algorithm, with many original optimizations, that is used in several of Alpage's NLP tools, including the pre-processing chain SXPipe and the LFG deep parser SXLFG. This implementation of the Earley algorithm has been recently extended to handle probabilistic CFG (PCFG), by taking into account probabilities both during parsing (beam) and after parsing (n-best computation). SYNTAX 6.0 also includes parsers for various contextual formalisms, including a parser for Range Concatenation Grammars (RCG) that can be used among others for TAG and MC-TAG parsing.

Direct NLP users of SYNTAX for NLP, outside Alpage, include Alexis Nasr (Marseilles) and other members of the (now closed) SEQUOIA ANR project, Owen Rambow and co-workers at Columbia University (New York), as well as (indirectly) all SXPipe and/or SXLFG users. The project-team VASY (Inria Rhône-Alpes) is one of SYNTAX' user for non-NLP applications.

5.2. DyALog

Participant: Éric Villemonte de La Clergerie [maintainer].

DYALOG *on Inria GForge: http://dyalog.gforge.inria.fr/*

DYALOG provides an environment to compile and execute grammars and logic programs. It is essentially based on the notion of tabulation, i.e. of sharing computations by tabulating traces of them. DYALOG is mainly used to build parsers for Natural Language Processing (NLP). It may nevertheless be used as a replacement for traditional PROLOG systems in the context of highly ambiguous applications where sub-computations can be shared.

The current release **1.13.0** of DYALOG is freely available by FTP under an open source license and runs on Linux platforms for x86 and architectures and on Mac OS intel (both 32 and 64bits architectures).

The current release handles logic programs, DCGs (*Definite Clause Grammars*), FTAGs (*Feature Tree Adjoining Grammars*), FTIGs (*Feature Tree Insertion Grammars*) and XRCGs (*Range Concatenation Grammars* with logic arguments). Several extensions have been added to most of these formalisms such as intersection, Kleene star, and interleave operators. Typed Feature Structures (TFS) as well as finite domains may be used for writing more compact and declarative grammars [101].

C libraries can be used from within DYALOG to import APIs (`mysql`, `libxml`, `sqlite`, ...).

DYALOG is largely used within ALPAGE to build parsers but also derivative softwares, such as a compiler of Meta-Grammars (cf. 5.3). It has also been used for building FRMG, a parser from a large coverage French TIG/TAG grammar derived from a Meta-Grammar. This parser has been used for the Parsing Evaluation campaign EASy, the two Passage campaigns (Dec. 2007 and Nov. 2009), cf. [99], [100], and very large amount of data (700 millions of words) in the SCRIBO project. New results concerning FRMG are described in 6.11 .

A new statistical dependency parser, based on a shift-reduce algorithm, was also developed in 2013 within the DYALOG system (see 6.12).

DYALOG and other companion modules are available on Inria GForge.

5.3. Tools and resources for Meta-Grammars

Participant: Éric Villemonte de La Clergerie [maintainer].

mgcomp, MGTOOLS, and FRMG on Inria GForge: http://mgkit.gforge.inria.fr/

DYALOG (cf. 5.2) has been used to implement `mgcomp`, Meta-Grammar compiler. Starting from an XML representation of a MG, `mgcomp` produces an XML representation of its TAG expansion.

The current version **1.5.0** is freely available by FTP under an open source license. It is used within ALPAGE and (occasionally) at LORIA (Nancy) and at University of Pennsylvania.

The current version adds the notion of namespace, to get more compact and less error-prone meta-grammars. It also provides other extensions of the standard notion of Meta-Grammar in order to generate very compact TAG grammars. These extensions include the notion of *guarded nodes*, i.e. nodes whose existence and non-existence depend on the truth value of a guard, and the use of the regular operators provided by DYALOG on nodes, namely disjunction, interleaving and Kleene star. The current release provides a dump/restore mechanism for faster compilations on incremental changes of a meta-grammars.

The current version of `mgcomp` has been used to compile a wide coverage Meta-Grammar FRMG (version 2.0.1) to get a grammar of around 200 TAG trees [12]. Without the use of guarded nodes and regular operators, this grammar would have more than several thousand trees and would be almost intractable. FRMG has been packaged and is freely available.

To ease the design of meta-grammars, a set of tools have been implemented, mostly by Éric Villemonte De La Clergerie, and collected in MGTOOLS (version **2.2.2**). This package includes a converter from a compact format to a XML pivot format, an Emacs mode for the compact and XML formats, a graphical viewer interacting with Emacs and XSLT stylesheets to derive HTML views.

The various tools on Metagrammars are available on Inria GForge. FRMG is used directly or indirectly (through a Web service or by requiring parsed corpora) by several people and actions (ANR Rhapsodie, ANR Chronoline, ...)

5.4. The Bonsai PCFG-LA parser

Participants: Marie-Hélène Candito [correspondant], Djamé Seddah, Benoît Crabbé.

Web page:
http://alpage.inria.fr/statgram/frdep/fr_stat_dep_parsing.html

Alpage has developed as support of the research papers [60], [53], [54], [11] a statistical parser for French, named Bonsai, trained on the French Treebank. This parser provides both a phrase structure and a projective dependency structure specified in [4] as output. This parser operates sequentially: (1) it first outputs a phrase structure analysis of sentences reusing the Berkeley implementation of a PCFG-LA trained on French by Alpage (2) it applies on the resulting phrase structure trees a process of conversion to dependency parses using a combination of heuristics and classifiers trained on the French treebank. The parser currently outputs several well known formats such as Penn treebank phrase structure trees, Xerox like triples and CONLL-like format for dependencies. The parsers also comes with basic preprocessing facilities allowing to perform elementary sentence segmentation and word tokenisation, allowing in theory to process unrestricted text. However it is believed to perform better on newspaper-like text. See 6.12 for recent work and results involving Bonsai.

The parser is available under a GPL license.

5.5. The MICA parser

Participants: Benoît Sagot [correspondant], Pierre Boullier.

Web page:
http://mica.lif.univ-mrs.fr/

MICA (Marseille-Inria-Columbia- AT&T) is a freely available dependency parser [48] currently trained on English and Arabic data, developed in collaboration with Owen Rambow and Daniel Bauer (Columbia University) and Srinivas Bangalore (AT&T). MICA has several key characteristics that make it appealing to researchers in NLP who need an off-the-shelf parser, based on Probabilistic Tree Insertion Grammars and on the SYNTAX system. MICA is fast (450 words per second plus 6 seconds initialization on a standard high-end machine) and has close to state-of-the-art performance (87.6% unlabeled dependency accuracy on the Penn Treebank).

MICA consists of two processes: the supertagger, which associates tags representing rich syntactic information with the input word sequence, and the actual parser, based on the Inria SYNTAX system, which derives the syntactic structure from the n-best chosen supertags. Only the supertagger uses lexical information, the parser only sees the supertag hypotheses.

MICA returns n-best parses for arbitrary n; parse trees are associated with probabilities. A packed forest can also be returned.

5.6. Alpage's linguistic workbench, including SxPipe

Participants: Benoît Sagot [correspondant], Rosa Stern, Marion Baranes, Damien Nouvel, Virginie Mouilleron, Pierre Boullier, Éric Villemonte de La Clergerie.

See also the web page http://lingwb.gforge.inria.fr/.

Alpage's linguistic workbench is a set of packages for corpus processing and parsing. Among these packages, the SXPipe package is of a particular importance.

SXPipe [80] is a modular and customizable chain aimed to apply to raw corpora a cascade of surface processing steps. It is used

- as a preliminary step before Alpage's parsers (e.g., FRMG);
- for surface processing (named entities recognition, text normalization, unknown word extraction and processing...).

Developed for French and for other languages, SXPipe includes, among others, various named entities recognition modules in raw text, a sentence segmenter and tokenizer, a spelling corrector and compound words recognizer, and an original context-free patterns recognizer, used by several specialized grammars (numbers, impersonal constructions, quotations...). In 2012, SXPipe has received a renewed attention in four directions:

- Support of new languages, and most notably German (although this is still at a very preliminary stage of development;
- Analysis of unknown words, in particular in the context of the ANR project EDyLex and of the collaboration with *viavoo*; this involves in particular (i) new tools for the automatic pre-classification of unknown words (acronyms, loan words...) (ii) new morphological analysis tools, most notably automatic tools for constructional morphology (both derivational and compositional), following the results of dedicated corpus-based studies (see 6.2 for new results);
- Development of new local grammars for detecting new types of entities and improvement of existing ones, in the context of the PACTE project (see 6.7 for new results).

5.7. MElt

Participants: Benoît Sagot [correspondant], Pierre Magistry.

MElt is a part-of-speech tagger, initially developed in collaboration with Pascal Denis (Magnet, Inria — then at Alpage), which was trained for French (on the French TreeBank and coupled with the Le*fff*), also trained on English [63], Spanish [69], Italian [94], German, Dutch, Polish, Kurmanji Kurdish [104] and Persian [89], [90]. It is state-of-the-art for French.

It is now able to handle noisy corpora (French and English only).

MElt also includes a lemmatization post-processing step.

A specific effort has been made towards the usability of MElt by linguists. In particular, a training session has been organized, and a user guide has been written.

Moreover, a preliminary version of MElt which accepts input DAGs has been developed.

MElt is distributed freely as a part of the Alpage linguistic workbench.

5.8. The Alexina framework: the Lefff syntactic lexicon, the Aleda entity database and other Alexina resources

Participants: Benoît Sagot [correspondant], Laurence Danlos.

See also the web page http://gforge.inria.fr/projects/alexina/.

Alexina is Alpage's Alexina framework for the acquisition and modeling of morphological and syntactic lexical information. The first and most advanced lexical resource developed in this framework is the Le*fff*, a morphological and syntactic lexicon for French.

Historically, the Le*fff* 1 was a freely available French morphological lexicon for verbs that has been automatically extracted from a very large corpus. Since version 2, the Le*fff* covers all grammatical categories (not just verbs) and includes syntactic information (such as subcategorization frames); Alpage's tools, including Alpage's parsers, rely on the Le*fff*. The version 3 of the Le*fff*, which has been released in 2008, improves the linguistic relevance and the interoperability with other lexical models.

Other Alexina lexicons exist, at various stages of development, in particular for Spanish (the Le*ff*e), Polish, Slovak, English, Galician, Persian, Kurdish, Italian,German, as well as for Latin verbs and a subset of Maltese and Khaling verbs. These lexicons are used in various tools, including instances of the MElt POS-tagger, and for studies in quantitative morphology.

Alexina also hosts *Aleda* [98], [88] a large-scale entity database currently developed for French but under development for English, Spanish and German, extracted automatically from Wikipedia and Geonames. It is used among others in the SXPipe processing chain and its NP named entity recognition, as well as in the NOMOS named entity linking system.

5.9. The free French wordnet WOLF

Participants: Benoît Sagot [correspondant], Sarah Beniamine.

The WOLF (Wordnet Libre du Français) is a wordnet for French, i.e., a lexical semantic database. The development of WOLF started in 2008 [82], [83]. At this time, we focused on benefiting from available resources of three different types: general and domain-specific bilingual dictionaries, multilingual parallel corpora and Wiki resources (Wikipedia and Wiktionaries). This work was achieved in a large part in collaboration with Darja Fišer (University of Ljubljana, Slovenia), in parallel with the development of a free Slovene wordnet, sloWNet. However, it was also impacted by specific collaborations, e.g., on adverbial synsets [84].

In 2013, a beta version of a new version of WOLF (version 1.0b2) was published, which integrates and extends the various efforts performed and published somewhat independently in 2012.

The WOLF is freely available under the Cecill-C license. It has already been used in various experiments, within and outside Alpage.

5.10. OGRE (Optimized Graph Rewriting Engine)

Participants: Corentin Ribeyre [correspondant], Djamé Seddah, Éric Villemonte de La Clergerie, Marie-Hélène Candito.

OGRE (Optimized Graph Rewriting Engine) is a graph rewriting system specifically designed for manipulating linguistic trees and graphs [78]. It relies on a rule specification language for expressing graph rewriting patterns. The transformation is performed in two steps:

1. First, the system performs simple transformations following the rewriting patterns;
2. Second, constraints can be applied on edges, which applies transformations depending on their environment that are propagated while all constraints are satisfied.

The system has been designd for the analysis and manipulation of attributed oriented and multi-relational graphs.

Web site: http://www.corentinribeyre.fr/projects/view/OGRE

5.11. Automatic construction of distributional thesauri

Participants: Marie-Hélène Candito [correspondant], Enrique Henestroza Anguiano.

FREDIST is a freely-available (LGPL license) Python package that implements methods for the automatic construction of distributional thesauri.

We have implemented the context relation approach to distributional similarity, with various context relation types and different options for weight and measure functions to calculate distributional similarity between words. Additionally, FREDIST is highly flexible, with parameters including: context relation type(s), weight function, measure function, term frequency thresholds, part-of-speech restrictions, filtering of numerical terms, etc.

Distributional thesauri for French are also available, one each for adjectives, adverbs, common nouns, and verbs. They have been constructed with FREDIST and use the best settings obtained in an evaluation. We use the *L'Est Republicain* corpus (125 million words), *Agence France-Presse* newswire dispatches (125 million words) and a full dump of the French Wikipedia (200 million words), for a total of 450 million words of text.

5.12. Tools and resources for time processing

Participant: Laurence Danlos [correspondant].

Alpage developed the *French TimeBank*, a freely-available corpus annotated with ISO-TimeML-compliant temporal information (dates, events and relations between events) [1].

5.13. LexViz

Participants: Mikael Morardo [maintainer], Éric Villemonte de La Clergerie.

In the context of the industrial collaboration of ALPAGE with the company Lingua & Machina, we have extended their WEB plateform Libellex with a new component used to visualize and collaboratively validate lexical resources. In particular, this extension is used to manage terminological lists and lexical networks. The implemented graph-based representation has proved to be intuitive and quite useful for navigating in such large lexical resources (on the order to 10K to 100K entries).

5.14. Mgwiki

Participants: Paul Bui Quang [maintainer], Éric Villemonte de La Clergerie.

In the context of Inria ADT Mgwiki, Paul Bui Quang has developed a linguistic wiki that may used to discuss linguistic phenomena with the possibility to add annotated illustrative sentences. The work is essentially devoted to the construction of an instance for documenting and discussing FRMG, with the annotations of the sentences automatically provided by parsing them with FRMG. This instance also offers the possibility to parse small corpora with FRMG and an interface of visualization of the results.

Another instance was deployed for managing the annotation guide for the Deep version of the Sequoia treebank, confirming the potential of the notion of linguistic wiki

5.15. NewsProcess

Participants: Éric Villemonte de La Clergerie [maintainer], Damien Nouvel.

NewsProcess is an HTTP-based service that may be used to process AFP news through the Alpage Processing Chain, in order to extract information, in particular citations. The chain has been completed to track the emergence of new words in the news.

In the context on ANR EdyLex, a new version of NewsProcess has been designed for processing AFP news wires and extracting information about unknown words (see 6.2)

5.16. System EasyRef

Participant: Éric Villemonte de La Clergerie [maintainer].

A collaborative WEB service EASYREF has been developed, in the context of ANR action Passage, to handle syntactically annotated corpora. EASYREF may be used to view annotated corpus, in both EASY or PASSAGE formats. The annotations may be created and modified. Bug reports may be emitted. The annotations may be imported and exported. The system provides standard user right management. The interface has been designed with the objectives to be intuitive and to speed edition.

EASYREF relies on an Model View Controller design, implemented with the Perl Catalyst framework. It exploits WEB 2.0 technologies (i.e. AJAX and JavaScript).

Version 2 has been used by ELDA and LIMSI to annotate a new corpus of several thousands words for the former ANR projectPASSAGE.

EASYREF is maintained under Inria GForge.

SMIS Project-Team

5. Software and Platforms

5.1. Introduction

In our research domain, developing software prototypes is mandatory to validate research solutions and is an important vector for publications, demonstrations at conferences and exhibitions as well as for cooperations with industry. This prototyping task is however difficult because it requires specialized hardware platforms (e.g., new generations of smart tokens), themselves sometimes at an early stage of development.

For a decade, we have developed successive prototypes addressing different application domains, introducing different technical challenges and relying on different hardware platforms. PicoDBMS was our first attempt to design a full-fledged DBMS embedded in a smart card [39] [27]. Chip-Secured Data Access (C-SDA) embedded a reduced SQL query engine and access right controller in a secure chip and acted as an incorruptible mediator between a client and an untrusted server hosting encrypted data [34]. Chip-Secured XML Access (C-SXA) was an XML-based access rights controller embedded in a smart card [35]. Prototypes of C-SXA have been the recipient of the e-gate open 2004 Silver Award and SIMagine 2005 Gold award, two renowned international software contests. The next subsections detail the two prototypes we are focusing on today.

5.2. PlugDB engine

Participants: Nicolas Anciaux [correspondent], Luc Bouganim, Philippe Pucheral, Shaoyi Yin, Yanli Guo, Lionel Le Folgoc, Alexei Troussov.

More than a stand-alone prototype, PlugDB is part of a complete architecture dedicated to a secure and ubiquitous management of personal data. PlugDB aims at providing an alternative to a systematic centralization of personal data. To meet this objective, the PlugDB architecture lies on a new kind of hardware device called Secure Portable Token (SPT). Roughly speaking, a SPT combines a secure microcontroller (similar to a smart card chip) with a large external Flash memory (Gigabyte sized). The SPT can host data on Flash (e.g., a personal folder) and safely run code embedded in the secure microcontroller. PlugDB engine is the cornerstone of this embedded code. PlugDB engine manages the database on Flash (tackling the peculiarities of NAND Flash storage), enforces the access control policy defined on this database, protects the data at rest against piracy and tampering, executes queries (tackling low RAM constraint) and ensures transaction atomicity. Part of the on-board data can be replicated on a server (then synchronized) and shared among a restricted circle of trusted parties through crypto-protected interactions. PlugDB engine has been registered at APP (Agence de Protection des Programmes) in 2009 [28] and a new version is registered each year. The underlying Flash-based indexing system has also been patented by Inria and Gemalto [40]. It has been demonstrated in a dozen of national and international events including JavaOne and SIGMOD. It is being experimented in the field to implement a secure and portable medical-social folder helping the coordination of medical care and social services provided at home to dependent people. The next step in our agenda is to put this software in open-source so that students and communities of developers can complement it and develop innovative privacy-by-design applications. In 2012, we have ported PlugDB-engine on a new hardware platform to 1) become completely independent from Gemalto, 2) have a plug-and-play implementation on Android, 3) serve as a basement to port it on other custom hardware implementations. We have already discussed with hardware companies located in "Ile-de-France" to produce new hardware tokens to host future versions of PlugDB-engine. Link: http://www-smis.inria.fr/_DMSP/home.php.

5.3. uFLIP Benchmark

Participants: Luc Bouganim [correspondent], Philippe Bonnet, Bjorn Jónsson, Lionel Le Folgoc.

It is amazingly easy to produce meaningless results when measuring flash devices, partly because of the peculiarity of flash memory, but primarily because their behavior is determined by layers of complex, proprietary, and undocumented software and hardware. uFLIP is a component benchmark for measuring the response time distribution of flash IO patterns, defined as the distribution of IOs in space and time. uFLIP includes a benchmarking methodology which takes into account the particular characteristics of flash devices. The source code of uFLIP, available on the web (700 downloads, 4000 distinct visitors), was registered at APP in 2009 [32]. It has been demonstrated at SIGMOD.

Link: http://www.uflip.org.

WILLOW Project-Team

5. Software and Platforms

5.1. SPArse Modeling Software (SPAMS)

SPAMS v2.4 was released as open-source software in May 2013 (v1.0 was released in September 2009, v2.0 in November 2010). It is an optimization toolbox implementing algorithms to address various machine learning and signal processing problems involving

- Dictionary learning and matrix factorization (NMF, sparse PCA, ...)
- Solving sparse decomposition problems with LARS, coordinate descent, OMP, SOMP, proximal methods
- Solving structured sparse decomposition problems (ℓ_1/ℓ_2, ℓ_1/ℓ_∞, sparse group lasso, tree-structured regularization, structured sparsity with overlapping groups,...).

The software and its documentation are available at http://www.di.ens.fr/willow/SPAMS/.

5.2. Local dense and sparse space-time features

This is a package with Linux binaries implementing extraction of local space-time features in video. We are preparing a new release of the code implementing highly-efficient video descriptors described in Section 6.4.3 . Previous version of the package was released in January 2011. The code supports feature extraction at Harris3D points, on a dense space-time grid as well as at user-supplied space-time locations. The package is publicly available at http://www.di.ens.fr/~laptev/download/stip-2.0-linux.zip.

5.3. Automatic Mining of Visual Architectural Elements

The code on automatic mining of visual architectural elements (v4.5) described in (Doersch *et al.* SIGGRAPH 2012) has been publicly released online in January 2013 (earlier version v4.3 was released in December 2012 and v3.0 was released in September 2012) at http://graphics.cs.cmu.edu/projects/whatMakesParis/paris_sigg_release_v4.5.tar.gz.

5.4. Joint learning of actors and actions in video

This is a package of Matlab code implementing the multi-view face processing pipeline and joint learning of actors and actions in movies described in (Bojanowski *et al.* ICCV 2013 [2]. The package was last updated in December 2013 and is available at http://www.di.ens.fr/willow/research/actoraction/.

5.5. Visual Place Recognition with Repetitive Structures

Open-source release of the software package for visual localization in urban environments has been made publicly available. The software package implements newly developed method [9] for representing visual data containing repetitive structures (such as building facades or fences), which often occur in urban environments and present significant challenge for current image matching methods. The software is available at http://www.di.ens.fr/willow/research/repttile/download/repttile_demo_ver02.zip.

IMARA Project-Team

5. Software and Platforms

5.1. MELOSYM

Participants: Fawzi Nashashibi [correspondant], Benjamin Lefaudeux, Paulo Lopes Resende.

MELOSYM is the acronym for "**M**odélisation de l'**E**nvironnement et **LO**calisation en temps réel pour un **SY**stème **M**obile autonome ou pas, fondé sur des données du capteur laser". This is a SLAM based algorithm for the environment mapping and vehicle localization in real-time using laser data. The particularity of the algorithm is its hierarchical approach that improves the accuracy of the system and speeds up the computations. Version 3 is under edition. It runs now in standalone mode without the use of RTMaps software libraries.

- Version: V2

5.2. Stereoloc-3D

Participants: Benjamin Lefaudeux, Fawzi Nashashibi [correspondant].

This software is a stereovision based system capable of performing a vehicle or robot ego-localization and 3D environment mapping in real-time. It has also the capability to ensure mobile objects detection and tracking. A new updated version has been released and tested on a mobile platform.

- Version: V1

5.3. Fuzzy logic tool

Participant: Joshué Pérez Rastelli [correspondant].

A fuzzy logic module has been implemented to translate human knowledge to driverless control processes, considering risk/warning situation. Fuzzy logic techniques have been widely implemented in different industrial process in the last decade. For this reason, many libraries, mainly developed in C++, are easily found in the literature. The goal is to achieve the autonomous driving of the vehicle using simple sentences defined in a rule base. Then, it is just necessary to define the input and output membership functions. Two modules based on fuzzy logic libraries were created. One of them was developed in order to compare the classic controller of a previous work with a fuzzy controller to improve the lateral control tracking previously developed. Moreover, another module to warn speed references at intersections with traffic lights was done in the framework of the project CoDrive. The idea is that the vehicle is able to know at which speed it must travel to avoid abrupt braking and save fuel.

5.4. Dynamic path generation

Participant: Joshué Pérez Rastelli [correspondant].

An algorithm for dynamic path generation in urban environments is presented, taking into account structural and sudden changes in straight and bend segments (e.g. roundabouts and intersections). The results present some improvements in path generation (previously hand plotted) considering parametric equations and continuous-curvature algorithms, which guarantees a comfortable lateral acceleration. This work is focused in a smooth and safe path generation using road and obstacle detection information. Finally, some simulation results show a good performance of the algorithm using different ranges of urban curves.

5.5. V2ProVu

Participants: Pierre Merdrignac, Oyunchimeg Shagdar [correspondant].

A Java-based software is developed to enable direct Wi-Fi communications between devices, especially between vehicle on-board communication devices and pedestrian hand-held devices (e.g., tablets). The software includes an algorithm that calculates vehicle-to-pedestrian collision risk and GUI, for hazard alarming.

5.6. Network Selector

Participant: Oyunchimeg Shagdar [correspondant].

An OSGi based software is developed under the scope of SCORE@F project. The software has the functionality of switching between Geo- and IP-networks in vehicular communications allowing e.g., Cooperative Awareness Messages (CAM) as well as Decentralized Event Notification Messages (DENM) being able distributed over one or both of the Geo- and IP-networks.

5.7. FAC-CM

Participant: Manabu Tsukada [correspondant].

An OSGi based software is developed under the scope of SCORE@F project. The software allows information exchange between Facilities and Management entities of ITS stations (e.g., vehicle on-board communication device).