RESEARCH CENTER
Nancy - Grand Est

FIELD

Activity Report 2012

Section Software

Edition: 2013-04-24

CAMUS Team

5. Software

5.1. PolyLib

PolyLib [8] is a C library of polyhedral functions, that can manipulate unions of rational polyhedra of any dimension, through the following operations: intersection, difference, union, convex hull, simplify, image and preimage. It was the first to provide an implementation of the computation of parametric vertices of a parametric polyhedron, and the computation of an Ehrhart polynomial (expressing the number of integer points contained in a parametric polytope) based on an interpolation method.

It is used by an important community of researchers (in France and the rest of the world) in the area of compilation and optimization using the polyhedral model. Vincent Loechner is the maintainer of this software. It is distributed under GNU General Public License version 3 or later, and it has a Debian package maintained by Serge Guelton (Symbiose Projet, IRISA).

5.2. ZPolyTrans

ZPolyTrans [9] is a C library and a set of executables, that permits to compute the integer transformation of a union of parametric $\mathbb{Z}$-polyhedra (the intersection between lattices and parametric polyhedra), as a union of parametric $\mathbb{Z}$-polyhedra. The number of integer points of the result can also be computed. It is build upon PolyLib and Barvinok library. This work is based on some theoretical results obtained by Rachid Seghir and Vincent Loechner [15].

It allows for example to compute the number of solutions of a Presburger formula by eliminating existential integer variables, or to compute the number of different data accessed by some array accesses contained in an affine parametric loop nest.

The authors of this software are Rachid Seghir (Univ. Batna, Algeria) and Vincent Loechner. It is distributed under GNU General Public License version 3 or later.

5.3. NLR

Participant: Alain Ketterlin.

We have developed a program implementing our loop-nest recognition algorithm, detailed in [7]. This standalone, filter-like application takes as input a raw trace and builds a sequence of loop nests that, when executed, reproduce the trace. It is also able to predict forthcoming values at an arbitrary distance in the future. Its simple, text-based input format makes it applicable to all kinds of data. These data can take the form of simple numeric values, or have more elaborate structure, and can include symbols. The program is written in standard ANSI C. The code can also be used as a library.

We have used this code to evaluate the compression potential of loop nest recognition on memory address traces, with very good results. We have also shown that the predictive power of our model is competitive with other models on average.

The software is available upon request to anybody interested in trying to apply loop nest recognition. It has been distributed to a dozen of colleagues around the world. In particular, it has been used by Andres Charif-Rubial for his PhD work (Université de Versailles Saint-Quentin en Yvelines), and is now included in a released tool called MAQAO (http://www.maqao.org). Our code is also used by Jean-Thomas ACQUAVIVA, at Commissariat à l'Énergie Atomique, for work on compressing instruction traces. These colleagues have slightly modified the code we gave them. We plan to release a stable version incorporating most of their changes in the near future. We also plan to change the license to avoid such drifts in the future.

[8] http://icps.u-strasbg.fr/PolyLib
[9] http://ZPolyTrans.gforge.inria.fr

5.4. Binary files decompiler

Participant: Alain Ketterlin.

Our research on efficient memory profiling has led us to develop a sophisticated decompiler. This tool analyzes x86-64 binary programs and libraries, and extracts various structured representations of the code. It works on a routine per routine basis, and first builds a loop hierarchy to characterize the overall structure of the algorithm. It then puts the code into Static Single Assignment (SSA) form to highlight the fine-grain data-flow between registers and memory. Building on these, it performs the following analyzes:

- All memory addresses are expressed as symbolic expressions involving specific versions of register contents, as well as loop counters. Loop counter definitions are recovered by resolving linearly incremented registers and memory cells, i.e., registers that act as induction variables.
- Most conditional branches are also expressed symbolically (with registers, memory contents, and loop counters). This captures the control-flow of the program, but also helps in defining what amounts to loop "trip-counts", even though our model is slightly more general, because it can represent any kind of iterative structure.

This tool embodies several passes that, as far as we know, do not exist in any existing similar tool. For instance, it is able to track data-flow through stack slots in most cases. It has been specially designed to extract a representation that can be useful in looking for parallel (or parallelizable) loops [45]. It is the basis of several of our studies.

Because binary program decompilation is especially useful to reduce the cost of memory profiling, our current implementation is based on the Pin binary instrumenter. It uses Pin's API to analyze binary code, and directly interfaces with the upper layers we have developed (e.g., program skeletonization, or minimal profiling). However, we have been careful to clearly decouple the various layers, and to not use any specific mechanism in designing the binary analysis component. Therefore, we believe that it could be ported with minimal effort, by using a binary file format extractor and a suitable binary code parser. It is also designed to abstract away the detailed instruction set, and should be easy to port (even though we have no practical experience in doing so).

We feel that such a tool could be useful to other researchers, because it makes binary code available under abstractions that have been traditionally available for source code only. If sufficient interest emerges, e.g., from the embedded systems community, or from researchers working on WCET, or from teams working on software security, we are willing to distribute and/or to help make it available under other environments.

5.5. Parwiz: a dynamic dependency analyser

Participant: Alain Ketterlin.

We have developed a dynamic dependence analyzer. Such a tool consumes the trace of memory (or object) accesses, and uses the program structure to list all the data dependences appearing during execution. Data dependences in turn are central to the search for parallel sections of code, with the search for parallel loops being only a particular case of the general problem. Most current works of these questions are either specific to a particular analysis (e.g., computing dependence densities to select code portions for thread-level speculation), or restricted to particular forms of parallelism (e.g., typically to fully parallel loops). Our tool tries to generalize existing approaches, and focuses on the program structures to provide helpful feedback either to a user (as some kind of "smart profiler"), or to a compiler (for feedback-directed compilation). For example, the tool is able to produce a dependence schema for a complete loop nest (instead of just a loop). It also targets irregular parallelism, for example analyzing a loop execution to estimate the expected gain of parallelization strategies like inspector-executor.

We have developed this tool in relation to our minimal profiling research project. However, the tool itself has been kept independent of our profiling infrastructure, getting data from it via a well-defined trace format. This intentional design decision has been motivated by our work on distinct execution environments: first on our usual x86-64 benchmark programs, and second on less regular, more often written in Java, real-world applications. The latter type of applications is likely the one that will most benefit from such tools, because their intrinsic execution environment does not offer enough structure to allow effective static analysis techniques. Parallelization efforts in this context will most likely rely on code annotations, or specific programming language constructs. Programmers will therefore need tools to help them choose between various constructs. Our tool has this ambition. We already have a working tool-chain for C/C++/Fortran programs (or any binary program). We are in the process of developing the necessary infrastructure to connect the dynamic dependence profiler to instrumented Java programs. Other managed execution environments could be targeted as well, e.g., Microsoft's .Net architecture, but we have no time and/or workforce to devote to such time-consuming engineering efforts.

5.6. VMAD software and LLVM

Participants: Alexandra Jimborean, Philippe Clauss, Jean-François Dollinger, Aravind Sukumaran-Rajam, Juan Manuel Martinez Caamaño.

For dynamic analysis and optimization of programs, we are developing a virtual machine called VMAD, and specific passes to the LLVM compiler suite, plus a modified Clang frontend. It is fully described in subsection 6.1 .

As the final result of Alexandra Jimborean's PhD thesis, the VMAD framework now handles speculative parallelization of loop nests by applying dynamically polyhedral code transformations. It is currently extended to handle even more advanced code transformations as tiling in particular, and also to handle codes whose memory behavior is not fully linear.

Alexandra Jimborean (PhD student), Matthieu Herrmann (former Master student), Luis Mastrangelo (former Master student), Juan Manuel Martinez Caamaño (Master student), Jean-François Dollinger (PhD student), Aravind Sukumaran-Rajam (PhD student) and Philippe Clauss are the main contributors of this software. It is not yet distributed.

5.7. Polyhedral prover

Participants: Nicolas Magaud, Julien Narboux, Éric Violard [correspondant].

polyhedral transformations, verified compiler

We are currently developing a formal proof of program transformations based on the polyhedral model. We use the CompCert verified compiler [51] as a framework. This tool is written in the specification language of Coq. It is connected to the activity described in section 6.5 .

CARAMEL Project-Team

5. Software

5.1. Introduction

A major part of the research done in the CARAMEL team is published within software. On the one hand, this enables everyone to check that the algorithms we develop are really efficient in practice; on the other hand, this gives other researchers — and us of course — basic software components on which they — and we — can build other applications.

5.2. GNU MPFR

Participant: Paul Zimmermann [contact].

GNU MPFR is one of the main pieces of software developed by the CARAMEL team. Since end 2006, with the departure of Vincent Lefèvre to ENS Lyon, it has become a joint project between CARAMEL and the ARÉNAIRE project-team (now AriC, INRIA Grenoble - Rhône-Alpes). GNU MPFR is a library for computing with arbitrary precision floating-point numbers, together with well-defined semantics, and is distributed under the LGPL license. All arithmetic operations are performed according to a rounding mode provided by the user, and all results are guaranteed correct to the last bit, according to the given rounding mode.

Several software systems use GNU MPFR, for example: the GCC and GFORTRAN compilers; the SAGE computer algebra system; the KDE calculator Abakus by Michael Pyne; CGAL (Computational Geometry Algorithms Library) developed by the Geometrica project-team (INRIA Sophia Antipolis - Méditerranée); Gappa, by Guillaume Melquiond; Sollya, by Sylvain Chevillard, Mioara Joldeş and Christoph Lauter; Genius Math Tool and the GEL language, by Jiri Lebl; Giac/Xcas, a free computer algebra system, by Bernard Parisse; the iRRAM exact arithmetic implementation from Norbert Müller (University of Trier, Germany); the Magma computational algebra system; and the Wcalc calculator by Kyle Wheeler.

The main development in 2012 is the release of version 3.1.1 (the "canard à l'orange" release) in July. With respect to version 3.1.0, this new version improves the reference manual, and fixes a few bugs. Also, a workshop was organized in June in Bordeaux, on the development of GNU MPFR and GNU MPC. In particular, the test coverage of GNU MPFR was improved.

5.3. GNU MPC

Participant: Paul Zimmermann [contact].

GNU MPC is a floating-point library for complex numbers, which is developed on top of the GNU MPFR library, and distributed under the LGPL license. It is co-written with Andreas Enge (LFANT project-team, INRIA Bordeaux - Sud-Ouest). A complex floating-point number is represented by $x + iy$, where x and y are real floating-point numbers, represented using the GNU MPFR library. The GNU MPC library provides correct rounding on both the real part x and the imaginary part y of any result. GNU MPC is used in particular in the TRIP celestial mechanics system developed at IMCCE (*Institut de Mécanique Céleste et de Calcul des Éphémérides*), and by the Magma and Sage computational number theory systems.

A new version, GNU MPC 1.0 (Fagus silvatica), was released in July 2012. Up from this release, GNU MPC is considered to be a mature library. Due to a security issue in automake, we had to release a bug-fix version 1.0.1 in September 2012.

5.4. GMP-ECM

Participants: Cyril Bouvier, Paul Zimmermann [contact].

GMP-ECM is a program to factor integers using the Elliptic Curve Method. Its efficiency comes both from the use of the GMP library, and from the implementation of state-of-the-art algorithms. GMP-ECM contains a library (LIBECM) in addition to the binary program (ECM). The binary program is distributed under GPL, while the library is distributed under LGPL, to allow its integration into other non-GPL software. The Magma computational number theory software and the SAGE computer algebra system both use LIBECM.

In January 2012, a new version 6.4 was released, followed by 6.4.1 and 6.4.2 in March, and 6.4.3 in June. Apart from bug fixes, and the fact that GMP-ECM is now distributed under the LGPL version 3, those new releases provide a new *-batch* option with faster Stage 1 code, and an improved tuning mechanism.

In February, Paul Leyland found a 43-digit factor using the GPU implementation of Stage 1 written by C. Bouvier, and in August, a new record prime of 79 digits was found by Sam Wagstaff (Purdue University) using GMP-ECM.

5.5. Finite fields

Participants: Pierrick Gaudry, Emmanuel Thomé [contact].

$\texttt{mp}\mathbb{F}_q$ is (yet another) library for computing in finite fields. The purpose of $\texttt{mp}\mathbb{F}_q$ is not to provide a software layer for accessing finite fields determined at runtime within a computer algebra system like Magma, but rather to give a very efficient, optimized code for computing in finite fields precisely known at *compile time*. $\texttt{mp}\mathbb{F}_q$ is not restricted to a finite field in particular, and can adapt to finite fields of any characteristic and any extension degree. However, one of the targets being the use in cryptology, $\texttt{mp}\mathbb{F}_q$ somehow focuses on prime fields and on fields of characteristic two.

$\texttt{mp}\mathbb{F}_q$'s ability to generate specialized code for desired finite fields differentiates this library from its competitors. The performance achieved is far superior. For example, $\texttt{mp}\mathbb{F}_q$ can be readily used to assess the throughput of an efficient software implementation of a given cryptosystem. Such an evaluation is the purpose of the "eBATS" benchmarking tool [1]. $\texttt{mp}\mathbb{F}_q$ entered this trend in 2007, establishing reference marks for fast elliptic curve cryptography: the authors improved over the fastest examples of key-sharing software in genus 1 and 2, both over binary fields and prime fields. These timings are now comparison references for other implementations [18].

The library's purpose being the *generation* of code rather than its execution, the working core of $\texttt{mp}\mathbb{F}_q$ consists of roughly 18,000 lines of Perl code, which generate most of the `C` code. $\texttt{mp}\mathbb{F}_q$ is distributed at http://mpfq.gforge.inria.fr/.

In 2012, $\texttt{mp}\mathbb{F}_q$ evolved somewhat, in order to do the required code generation needed for evolutions of CADO-NFS, notably in relation with linear algebra over prime fields. A new release is planned soon, once hindrances related to the license of some code fragments are dealt with.

5.6. gf2x

Participants: Pierrick Gaudry, Emmanuel Thomé [contact], Paul Zimmermann.

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

The current version of GF2X is 1.1, released in May 2012 under the GNU GPL. Since 2009, GF2X can be used as an auxiliary package for the widespread software library NTL, as of version 5.5.

An LGPL-licensed portion of GF2X is also part of the CADO-NFS software package.

[1] http://www.ecrypt.eu.org/ebats/

5.7. CADO-NFS

Participants: Cyril Bouvier, Jérémie Detrey, Alain Filbois, Pierrick Gaudry, Alexander Kruppa, Emmanuel Thomé [contact], Paul Zimmermann.

CADO-NFS is a program to factor integers using the Number Field Sieve algorithm (NFS), originally developped in the context of the ANR-CADO project (November 2006 to January 2010).

NFS is a complex algorithm which contains a large number of sub-algorithms. The implementation of all of them is now complete, but still leaves some places to be improved. Compared to existing implementations, the CADO-NFS implementation is already a reasonable player. Several factorizations have been completed using our implementations.

Since 2009, the source repository of CADO-NFS is publicly available for download. No new release was made in 2012, but several improvements have been made in the development version, with the help of Alain Filbois (SED engineer) and of Alexander Kruppa, recruited in October for a 2-year engineer contract.

Alain Filbois improved the *purge* program for filtering, by gaining a factor of about 5 in the input-output routines. Together with P. Zimmermann, he also wrote a special-purpose *clique removal* code for huge factorizations requiring out-of-core computing; this code has been used for a new filtering experiment on the relations collected for RSA-768 (not yet finished at the time of writing).

The *Objectif 1024* ADT started in 2012, with the recruitment of Alexander Kruppa as a engineer for 2 years. The four main objectives of this ADT are: (1) be able to use CADO-NFS routinely on clusters of 20 to 100 nodes, including on Amazon EC2; (2) develop precise tools to optimize parameters in the sieving phase; (3) develop more professional test mechanisms; (4) make two major releases of CADO-NFS, and advertize them on potential users.

Overall, CADO-NFS keeps improving its competitivity over alternative code bases. Improvements in CADO-NFS and new results obtained with CADO-NFS are described below.

CARTE Project-Team

5. Software

5.1. Morphus/MMDEX

MMDEX is a virus detector based on morphological analysis. It is composed of our own disassembler tool, on a graph transformer and a specific tree-automaton implementation. The tool is used in the EU-Fiware project and by some other partners (e.g. DAVFI project).
Written in C, 20k lines.
APP License, IDDN.FR.001.300033.000.R.P.2009.000.10000, 2009.

5.2. TraceSurfer

TraceSurfer is a self-modifying code analyzer coming with an IDA add-on. It works as a wave-builder. In the analysis of self-modifying programs, one basic task is indeed to separate parts of the code which are self-modifying into successive layers, called waves. TraceSurfer extracts waves from traces of program executions. Doing so drastically simplifies program verification.
Written in C, 5k lines.
Private licence.
http://code.google.com/p/tartetatintools/

5.3. CROCUS

CROCUS is a program interpretation synthetizer. Given a first order program (possibly written in OCAML), it outputs a quasi-interpretation based on max, addition and product. It is based on a random algorithm. The interpretation is actually a certificate for the program's complexity. Users are non academics (some artists).
Written in Java, 5k lines.
Private licence.

CASSIS Project-Team

5. Software

5.1. Protocol Verification Tools

Participants: Pierre-Cyrille Héam, Olga Kouchnarenko, Michaël Rusinowitch, Mathieu Turuani, Laurent Vigneron.

5.1.1. AVISPA

Cassis has been one of the 4 partners involved in the European project AVISPA, which has resulted in the distribution of a tool for automated verification of security protocols, named *AVISPA* Tool. It is freely available on the web [1] and it is well supported. The *AVISPA* Tool compares favourably to related systems in scope, effectiveness, and performance, by (i) providing a modular and expressive formal language for specifying security protocols and properties, and (ii) integrating 4 back-ends that implement automatic analysis techniques ranging from *protocol falsification* (by finding an attack on the input protocol) to *abstraction-based verification* methods for both finite and infinite numbers of sessions.

5.1.2. CL-AtSe

We develop, as a first back-end of *AVISPA*, *CL-AtSe*, a Constraint Logic based Attack Searcher for cryptographic protocols. The *CL-AtSe* approach to verification consists in a symbolic state exploration of the protocol execution, for a bounded number of sessions. This necessary restriction (for decidability, see [79]) allows *CL-AtSe* to be correct and complete, i.e., any attack found by *CL-AtSe* is a valid attack, and if no attack is found, then the protocol is secure for the given number of sessions. Each protocol step is represented by a constraint on the protocol state. These constraints are checked lazily for satisfiability, where satisfiability means reachability of the protocol state. *CL-AtSe* includes a proper handling of sets (operations and tests), choice points, specification of any attack states through a language for expressing secrecy, authentication, fairness, non-abuse freeness, advanced protocol simplifications and optimizations to reduce the problem complexity, and protocol analysis modulo the algebraic properties of cryptographic operators such as XOR (exclusive or) and Exp (modular exponentiation). The handling of XOR and Exp has required to implement an optimized version of the combination algorithm of Baader & Schulz [68] for solving unification problems in disjoint unions of arbitrary theories.

CL-AtSe has been successfully used [67] to analyse France Telecom R&D, Siemens AG, IETF, or Gemalto protocols in funded projects. It is also employed by external users, e.g., from the AVISPA's community. Moreover, *CL-AtSe* achieves very good analysis times, comparable and sometimes better than state-of-the art tools in the domain (see [82] for tool details and precise benchmarks).

Recently, Cl-Atse has been enhanced in various ways. As an official back-end for the Avantssar European Project, the tool's development followed the project's requirements for semantic and functionalities. In particular, the tool now fully supports the Aslan semantic, including support for Horn Clauses (for intruder-independent deductions, like e.g. management of credentials), improved support for LTL-based security properties, objects management w.r.t. a set semantic (instead of multiset by default), or smarter behavior in presence of ACM communication channels (default and preferred channel mode for Cl-Atse is CCM). While unofficial in Avantssar, the tracing option to target some specific traces during analysis has also been renewed w.r.t. the new modeling of transitions within the Aslan syntax. Also, tool support and bug corrections for all Avantssar's tools is now processed through a bugzilla server (see https://regis.scienze.univr.it/bugzilla/bugzilla-4.0.4/), and online analysis and orchestration are available on our team server (https://cassis.loria.fr). Then again, Cl-Atse now supports negative constraints on the intruder's knowledge. This support is correct and complete without algebraic operators (like Xor and Exp.), and implements in practice the assumptions and

[1] http://www.avispa-project.org

methods from [32]. This important improvement to the analysis algorithm in Cl-Atse allows us to find much more adequate orchestrations, and thus to reduce the orchestrator's processing times in a large scale. It was also used to model e.g. separation of duties.

5.2. Testing Tools

Participants: Fabrice Bouquet, Frédéric Dadeau, Philippe Paquelier, Kalou Cabrera.

5.2.1. Hydra

In December 2008, we have started the redevelopment of our original testing tools environment, with two objectives: first, refactoring the existing developments, and, second, providing an open platform aiming at gathering together the various developments, increasing the reusability of components. The resulting platform, named Hydra, is a Eclipse-like platform, based on Plug-ins architecture. Plug-ins can be of five kinds: *parser* is used to analyze source files and build an intermediate format representation of the source; *translator* is used to translate from a format to another or to a specific file; *service* denotes the application itself, i.e. the interface with the user; *library* denotes an internal service that can be used by a service, or by other libraries; *tool* encapsulates an external tool. The following services have been developed so far:

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

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

5.2.2. jMuHLPSL

jMuHLPSL [9] is a mutant generator tool that takes as input a verified HLPSL protocol, and computes mutants of this protocol by applying systematic mutation operators on its contents. The mutated protocol then has to be analyzed by a dedicated protocol analysis tool (here, the AVISPA tool-set). Three verdicts may then arise. The protocol can still be *safe*, after the mutation, this means that the protocol is not sensitive to the realistic "fault" represented by the considered mutation. This information can be used to inform the protocol designers of the robustness of the protocol w.r.t. potential implementation choices, etc. The protocol can also become *incoherent*, meaning that the mutation introduced a functional failure that prevents the protocol from being executed entirely (one of the participants remains blocked in a given non-final state). The protocol can finally become *unsafe* when the mutation introduces a security flaw that can be exploited by an attacker. In this case, the AVISPA tool-set is able to compute an attack-trace, that represents a test case for the implementation of the protocol. If the attack can be replayed entirely, then the protocol is not safe. If the attack can not be replayed then the implementation does not contain the error introduced in the original protocol.

The tool is written in Java, and it is freely available at: http://disc.univ-fcomte.fr/home/~fdadeau/tools/jMuHLPSL.jar.

5.3. Collaborative Tools

Participants: Abdessamad Imine, Asma Cherif.

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

- **P2PEdit.** This prototype is implemented in Java and supports the collaborative editing of HTML pages and it is deployed on P2P JXTA platform [2]. In our prototype, a user can create a HTML page from scratch by opening a new collaboration group. Other users (peers) may join the group to participate in HTML page editing, as they may leave this group at any time. Each user can dynamically add and remove different authorizations for accessing to the shared document according the contribution and the competence of users participating in the group. Using JXTA platform, users exchange their operations in real-time in order to support WYSIWIS (What You See Is What I See) principle. Furthermore, the shared HTML document and its authorization policy are replicated at the local memory of each user. To deal with latency and dynamic access changes, an optimistic access control technique is used where enforcement of authorizations is retroactive.
- **P2PCalendar.** To extend our collaboration and access control models to mobile devices, we implemented a shared calendar on iPhone OS which is decentralized and scalable (i.e. it can be used over both P2P and ad-hoc networks). This application aims to make a collaborative calendar where users can simultaneously modify events (or appointements) and control access on events. The access rights are determined by the owner of an event. The owner decides who is allowed to access the event and what privileges they have. Likewise to our previous tool, the calendar and its authorization policy are replicated at every mobile device.

5.4. Other Tools

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

[2] http://www.sun.com/software/jxta/

PAREO Project-Team

5. Software

5.1. ATerm

Participant: Pierre-Etienne Moreau [correspondant].

ATerm (short for Annotated Term) is an abstract data type designed for the exchange of tree-like data structures between distributed applications.

The ATerm library forms a comprehensive procedural interface which enables creation and manipulation of ATerms in C and Java. The ATerm implementation is based on maximal subterm sharing and automatic garbage collection.

A binary exchange format for the concise representation of ATerms (sharing preserved) allows the fast exchange of ATerms between applications. In a typical application—parse trees which contain considerable redundant information—less than 2 bytes are needed to represent a node in memory, and less than 2 bits are needed to represent it in binary format. The implementation of ATerms scales up to the manipulation of ATerms in the giga-byte range.

The ATerm library provides a comprehensive interface in C and Java to handle the annotated term data-type in an efficient manner.

We are involved (with the CWI) in the implementation of the Java version, as well as in the garbage collector of the C version. The Java version of the ATerm library is used in particular by *Tom*.

The ATerm library is documented, maintained, and available at the following address: http://www.meta-environment.org/Meta-Environment/ATerms.

5.2. Tom

Participants: Jean-Christophe Bach, Christophe Calvès, Horatiu Cirstea, Pierre-Etienne Moreau [correspondant], Claudia Tavares.

Since 2002, we have developed a new system called *Tom* [33], presented in [17], [18]. This system consists of a pattern matching compiler which is particularly well-suited for programming various transformations on trees/terms and XML documents. Its design follows our experiments on the efficient compilation of rule-based systems [30]. The main originality of this system is to be language and data-structure independent. This means that the *Tom* technology can be used in a C, C++ or Java environment. The tool can be seen as a Yacc-like compiler translating patterns into executable pattern matching automata. Similarly to Yacc, when a match is found, the corresponding semantic action (a sequence of instructions written in the chosen underlying language) is triggered and executed. *Tom* supports sophisticated matching theories such as associative matching with neutral element (also known as list-matching). This kind of matching theory is particularly well-suited to perform list or XML based transformations for example.

In addition to the notion of *rule*, *Tom* offers a sophisticated way of controlling their application: a strategy language. Based on a clear semantics, this language allows to define classical traversal strategies such as *innermost*, *outermost*, *etc.*. Moreover, *Tom* provides an extension of pattern matching, called *anti-pattern matching*. This corresponds to a natural way to specify *complements* (*i.e.*what should not be there to fire a rule). *Tom* also supports the definition of cyclic graph data-structures, as well as matching algorithms and rewriting rules for term-graphs.

Tom is documented, maintained, and available at http://tom.loria.fr as well as at http://gforge.inria.fr/projects/tom.

TRIO Project-Team

5. Software

5.1. ANR Open-PEOPLE platform

Participants: Fabrice Vergnaud, Jérôme Vatrinet, Kévin Roussel, Olivier Zendra.

The aim of Open-PEOPLE is to provide a platform for estimating and optimizing the power and energy consumption of systems. The Open-PEOPLE project formally started in April 2009. Two systems administrator and software developers had been hired initially: Sophie Alexandre and Kévin Roussel. Another system administrator and software developer, Jonathan Ponroy, joined them in 2010 when he finished his work on the ANR MORE project where he worked previously. Sophie Alexandre contract ended in February 2011.

Since the beginning of the Open-PEOPLE project, we had made significant progress in setting up the infrastructure for the software part of the platform, for which Inria Nancy Grand Est is responsible. We had included new features to be able to fully integrate and test software developed as Eclipse plugins, relying on the Buckminster tool. We had also created a specific extension set for SVN and Hudson, called OPCIM (Open-PEOPLE Continuous Integration Mechanism). OPCIM had been registered at APP on 13/04/2010 with number IDDN.FR.001.150008.000.S.P.2010.000.10000.

Concerning the Open-PEOPLE platform itself, we had first tackled the high-level work, working with our partners on the definition of the requirements of the platform according to the needs of industry. We had then realized the specification work to define the global perimeter of our platform, according to the previous requirements. As part of this work had also been designed exchanges formats between the various tools. We had also designed at Inria Nancy Grand Est a Tools integration Protocol, which specified requirements for external tools to be integrated in our platform. All this design work had been materialized in several reports which were deliveries provided to ANR.

We had also designed and developed an authentication component (Eclipse plugin) for the platform, so as to be able to provide a unique, secured access gate to the platform to all the tools that are or shall be integrated into it.

We had also started and almost finished developing an Internet portal giving access and control to the Open-PEOPLE Hardware Platform, located at our partner's UBS in Lorient. Our portal features included user account management facilities, on the admin side, and on the user side, the ability to create, save, edit, reuse and of course submit jobs, make reservations for the hardware platform resources and get back tests results.

Finally, we had started working on two important parts of the software platform.

First, a way to unify the user experience despite the fact the platform federates several tools which were not developed to interact together. This implied an important and in-depth study of the wanted ergonomy for the platform, which involved taking into account both user needs and habits and the features of the available software tools.

The second work which had begun in 2011 was the design (then implementation) of the communications of between the various tools of the platform. This skeleton is a key part of our platform, and the quality of its design has a tremendous impact on its maintainability and its extensibility.

Note that the Open-PEOPLE project had been successfully evaluated on 14/09/2010 by ANR. Developments done during the first two years in the project are detailed in the 2009 and 2010 activity reports. In 2011, these developments had gone on.

We had continued the work to solidify our development platform supporting our work and that of our partners. We had produced a finer grained definition of the software platform functionalities, and a more precise definition of the tools integration protocol. We had worked towards the corresponding implementation documents, adding two new delivrables about the architecture of the software platform and the ergonomics of the software platform. For the latter, we had extensively interviewed user about ergonomics and designed several GUI mockups. We had progressed on the implementation of the software platform, especially with respect to the internet portal to remote-control the hardware platform. We had participated to the definition of the hardware platform and its functionalities, and participated actively to the work on the Specification document for HW / SW interfacing. We had provided the first concrete design and implementation of the HW/SW platform interfacing, with our implementation of the remote control portal for the HW platform. This remote control module had been completed in Fall 2011.

We had also participated to the work pertaining to basic components model homogenization, by reviewing this in the context of the software platform architecture and implementation, which had resulted in several incremental improvements of the underlying models. Finally, progressing towards the first release of the software part of the Open-PEOPLE platform, we had realized an ergonomic study for the consumption laws editors, with mockups and user interviews and validation. We had worked on the implementation of the editors for the consumption laws, which had required learning new environments and development tools (related to the EMF framework and the AADL, QUDV and MathML models). As a consequence, we had completed the implementation of the GUI and engine to create units and quantities. We had finalized the architecture needed to integrate external modules in the platform.

In 2012, this work went on. Basically, 2012 was the year of the concrete Open-PEOPLE platform, where all our efforts finally came to maturity. We thus completed global GUI of the Open-PEOPLE Software Platform. We performed the integration of various external tools and modules and the . We provided several improvements to the Remote Control Module providing access to the Hardware Platform. We finalized the implementation of consumption laws editors. We implemented the export and import functionality of Open-PEOPLE models. We created a new community-based website to allow sharing of Open-PEOPLE models.

We overall progressed as forecast in an iterative development and release schedule.

Version 1.0 (2012-04-06) was the first embodiment and public release of the Open-PEOPLE platform.

Version 1.1 (2012-06-12) added a default environment with pre-set Units, Quantities and AADL Property associations, asynchronous file uploads and downloads in the Remote Control Module, and better handling of big files (file size limit is now 4GB), and several bug fixes.

Version 1.3 (2012-09-27) changed internal mechanism of QUDV serialization (quantities, units and property associations), added version information to QUDV and Weaving meta-models, added internal builders to automatically generates QUDV and Weaving configuration files, added support of OSATE 2, improved UI reactivity (especially during file transfers), added progress bars for the remote control, and several bug fixes.

Version 1.4 (2012-10-25) added the Adele Graphical Editor, new OSATE 2 Snapshot, and several bugfixes.

Version 2.0 (2012-12-13) added RDALTE, AADL2SystemC, and a Standard environment with models and model sharing implemented (including a community sharing website), a new snapshot of OSATE2.

5.2. VITRAIL

Participants: Frédéric Diss, Pierre Caserta, Olivier Zendra.

The aim of the VITRAIL operation is to provide tools for the advanced and immersive visualization of programs. It partners with the University of Montréal, University of Montpellier and Pareo team of Inria Nancy Grand Est.

Last years, in VITRAIL, we had developed software to instrument and trace Java programs at the bytecode level. We then had developed an analysis tool able to exploit these traces to compute relevant software metrics. We had hired Damien Bodenes as software developer, and had begun the work on a prototype able to render a 3D world, symbolizing software, onto various visualization hardware, with the possibility to change the display metaphor. The main part of our development work had been in 2009 the choice and validation of the technology, and a first architecture. In 2010, the development had go on at a good pace, building on chosen technologies and architecture. This had brought new experience, and with the first actual runs of our platform, we had realized that with the Irrlicht platform we had chosen, we could reach unforeseeable problem when scaling up. We had thus decided to reverse our choice to the Ogre3D 3D engine at the beginning of 2010. Our development had then progressed steadily.

We had released in 2010 a first prototype of our platform, with all the underlying architecture, able to provide navigation features and interaction capacities limited to the driving of the navigation, as per our plans. This had included dual screen management.

Our first prototype, using 2 large 2D screens, with a city metaphor, had been demonstrated during the "Fête de la Science" in November 2010 and had received a lot of attention and enthusiasm from the general public. About 55 persons per day had visited our booth and got demonstrations.

We had also progressed significantly in our Java bytecode tracer, by improving its granularity, the completeness of the traced information, and its performance as well. We have a unique tool which is able to trace both program classes and JDK classes, at basic block level. In addition, it does so with a dynamic instrumentation of classes, which means there is no need to have an instrumented version of the class files on disk. This is very convenient, especially when changing machine of JVM, or when upgrading either the JDK or the program itself. In addition, the performance is good enough that the instrumented programs are still fully usable in an interactive way, without bothering the user. To the best of our knowledge, this is the only Java bytecode tracer that offers these features nowadays.

Our software development had lead to several registrations with APP:

- VITRAIL - Visualizer had been first registered on 29/12/2009 under number IDDN.FR.001.530021.000.S.P.2009.000.10000.
- VITRAIL - Tracer, was registered at APP on 20/09/2010 with number IDDN.FR.001.380001.000.S.P. 2010.000.10000.

In 2011, we acquired a workstation and three 30 inches computer screens, to be able to set up a "boxed 3D workstation", that would provide display in front and on both sides of the operator. This would constitute the next step in our experiments, by improving immersion with a larger field of vision (on the sides). The software developments to do this are ongoing. We also integrated a WiiMote interaction device to our system, but our experiments found that its spacial resolution was too poor for our needs.

We finally improved significantly our VITRAIL prototype in 2011, especially by designing and implementing a new representation for the relations between software (hence visual) elements, with limited clutter and the possibility to regroup links and see their direction.

In 2012, we continued working on the analysis of software, gathering statistics about polymorphism in Java programs, aiming at comparing various type analyses make statically (CHA, RTA, VTA) and the dynamic trace provided by (a) real execution(s). This work is going on and has not been published yet.

We also developed a public website for the VITRAIL project, which is going live these days.

VEGAS Project-Team

4. Software

4.1. QI: Quadrics Intersection

QI stands for "Quadrics Intersection". QI is the first exact, robust, efficient and usable implementation of an algorithm for parameterizing the intersection of two arbitrary quadrics, given in implicit form, with integer coefficients. This implementation is based on the parameterization method described in [10], [29], [30], [31] and represents the first complete and robust solution to what is perhaps the most basic problem of solid modeling by implicit curved surfaces.

QI is written in C++ and builds upon the LiDIA computational number theory library [24] bundled with the GMP multi-precision integer arithmetic [23]. QI can routinely compute parameterizations of quadrics having coefficients with up to 50 digits in less than 100 milliseconds on an average PC; see [10] for detailed benchmarks.

Our implementation consists of roughly 18,000 lines of source code. QI has being registered at the Agence pour la Protection des Programmes (APP). It is distributed under the free for non-commercial use Inria license and will be distributed under the QPL license in the next release. The implementation can also be queried via a web interface [25].

Since its official first release in June 2004, QI has been downloaded six times a month on average and it has been included in the geometric library EXACUS developed at the Max-Planck-Institut für Informatik (Saarbrücken, Germany). QI is also used in a broad range of applications; for instance, it is used in photochemistry for studying the interactions between potential energy surfaces, in computer vision for computing the image of conics seen by a catadioptric camera with a paraboloidal mirror, and in mathematics for computing flows of hypersurfaces of revolution based on constant-volume average curvature.

4.2. Isotop: Topology and Geometry of Planar Algebraic Curves

ISOTOP is a Maple software for computing the topology of an algebraic plane curve, that is, for computing an arrangement of polylines isotopic to the input curve. This problem is a necessary key step for computing arrangements of algebraic curves and has also applications for curve plotting. This software has been developed since 2007 in collaboration with F. Rouillier from Inria Paris - Rocquencourt. It is based on the method described in [28] which incorporates several improvements over previous methods. In particular, our approach does not require generic position.

Isotop is registered at the APP (June 15th 2011) with reference IDDN.FR.001.240007.000.S.P.2011.000.10000. This version is competitive with other implementations (such as ALCIX and INSULATE developed at MPII Saarbrücken, Germany and TOP developed at Santander Univ., Spain). It performs similarly for small-degree curves and performs significantly better for higher degrees, in particular when the curves are not in generic position.

We are currently working on an improved version integrating our new bivariate polynomial solver [27].

4.3. CGAL: Computational Geometry Algorithms Library

Born as a European project, CGAL (http://www.cgal.org) has become the standard library for computational geometry. It offers easy access to efficient and reliable geometric algorithms in the form of a C++ library. CGAL is used in various areas needing geometric computation, such as: computer graphics, scientific visualization, computer aided design and modeling, geographic information systems, molecular biology, medical imaging, robotics and motion planning, mesh generation, numerical methods...

In computational geometry, many problems lead to standard, though difficult, algebraic questions such as computing the real roots of a system of equations, computing the sign of a polynomial at the roots of a system, or determining the dimension of a set of solutions. we want to make state-of-the-art algebraic software more accessible to the computational geometry community, in particular, through the computational geometric library CGAL. On this line, we contributed a model of the *Univariate Algebraic Kernel* concept for algebraic computations [26] (see Sections 8.2.2 and 8.4). This CGAL package improves, for instance, the efficiency of the computation of arrangements of polynomial functions in CGAL [32]. We are currently developing a model of the *Bivariate Algebraic Kernel* based on our new bivariate polynomial solver [27]. This work is done in collaboration with F. Rouillier at Inria Paris - Rocquencourt and L. Peñaranda at the university of Athens.

4.4. Fast_polynomial: fast polynomial evaluation software

The library *fast_polynomial*[1] provides fast evaluation and composition of polynomials over several types of data. It is interfaced for the computer algebra system *sage*. This software is meant to be a first step toward a certified numerical software to compute the topology of algebraic curves and surfaces. It can also be useful as is and is submitted for integration in the computer algebra system *Sage*.

This software is focused on *fast online computation*, *multivariate evaluation*, *modularity*, and *efficiency*.

Fast online computation. The library is optimized for the evaluation of a polynomial on several point arguments given one after the other. The main motivation is numerical path tracking of algebraic curves, where a given polynomial criterion must be evaluated several thousands of times on different values arising along the path.

Multivariate evaluation. The library provides specialized fast evaluation of multivariate polynomials with several schemes, specialized for different types such as *mpz* big ints, *boost* intervals with hardware precision, *mpfi* intervals with any given precision, etc.

Modularity. The evaluation scheme can be easily changed and adapted to the user needs. Moreover, the code is designed to easily extend the library with specialization over new *C++* objects.

Efficiency. The library uses several tools and methods to provide high efficiency. First, the code uses templates, such that after the compilation of a polynomial for a specific type, the evaluation performance is equivalent to low-level evaluation. Locality is also taken into account: the memory footprint is minimized, such that an evaluation using the classical Hörner scheme will use $O(1)$ temporary objects and divide and conquer schemes will use $O(log(n))$ temporary objects, where n is the degree of the polynomial. Finally, divide and conquer schemes can be evaluated in parallel, using a number of threads provided by the user.

[1] http://trac.sagemath.org/sage_trac/ticket/13358

VERIDIS Project-Team

5. Software

5.1. The veriT solver

Participants: Rodrigo Castaño, David Déharbe, Pablo Federico Dobal, Pascal Fontaine [correspondent].

The veriT solver is an SMT (Satisfiability Modulo Theories) solver developed in cooperation with David Déharbe from the Federal University of Rio Grande do Norte in Natal, Brazil. The solver can handle large quantifier-free formulas containing uninterpreted predicates and functions, and arithmetic on integers and reals. It features a very efficient decision procedure for difference logic, as well as a simplex-based reasoner for full linear arithmetic. It also has some support for user-defined theories, quantifiers, and lambda-expressions. This allows users to easily express properties about concepts involving sets, relations, etc. The prover can produce an explicit proof trace when it is used as a decision procedure for quantifier-free formulas with uninterpreted symbols and arithmetic. To support the development of the tool, a regression platform using Inria's grid infrastructure is used; it allows us to extensively test the solver on thousands of benchmarks in a few minutes. The veriT solver is available as open source under the BSD license, and distributed through the web site http://www.veriT-solver.org.

Efforts in 2012 have been focused on efficiency, with various improvements and the redesign of the core solver. A preliminary prototype integrating Redlog for handling non-linear arithmetic showed encouraging results. Short term future works include improving the design, adding full support for non-linear arithmetic, and increasing efficiency.

We target applications where validation of formulas is crucial, such as the validation of TLA$^+$ and B specifications, and work together with the developers of the respective verification platforms to make veriT even more useful in practice. In 2012, we presented at ABZ [16] a plugin for Rodin using SMT solvers (and notably veriT) to discharge B proof obligations: on a large repository of industrial and academic cases, this SMT-based plugin decreased by 75% the number of proof obligations requiring human interactions, compared to the original B prover. See also section 8.1 for our work within the DeCert project.

For helping development within and around veriT, Pablo Federico Dobal has been hired for two years starting September 2012 as a young engineer supported by the Inria ADT program.

5.2. The TLA+ proof system

Participants: Stephan Merz [correspondent], Hernán-Pablo Vanzetto.

TLAPS, the TLA$^+$ proof system, is a platform for developing and mechanically verifying TLA$^+$ proofs. It is developed at the Joint MSR-Inria Centre. The TLA$^+$ proof language is declarative and based on standard mathematical logic; it supports hierarchical and non-linear proof construction and verification. TLAPS consists of a *proof manager* that interprets the proof language and generates a collection of proof obligations that are sent to *backend verifiers* that include theorem provers, proof assistants, SMT solvers, and decision procedures.

TLAPS is publically available at http://msr-inria.inria.fr/~doligez/tlaps/, it is distributed under a BSD-like license. It handles the non-temporal part of TLA$^+$ and can currently be used to prove safety, but not liveness properties. Its backends include a tableau prover for first-order logic, an encoding of TLA$^+$ in the proof assistant Isabelle, and a backend for interfacing with SMT solvers. The SMT backend has been improved significantly in 2012 and is now considered by users as the most useful backend prover for system verification. Version 1.0 of TLAPS was released in January 2012, followed by version 1.1 in November, and the system was presented at the conference FM 2012 [15].

CALVI Project-Team

5. Software

5.1. SeLaLib

Participants: Raphaël Blanchard, Edwin Chacon Golcher, Samuel De Santis, Aliou Diouf, Pierre Navaro, Morgane Bergot, Emmanuel Frénod, Philippe Helluy, Sever Hirstoaga, Michel Mehrenberger, Laurent Navoret, Eric Sonnendrücker.

Under the 'Fusion' large scale initiative, we have continued our work in the development of the ADT Selalib (the Semi-Lagrangian Library), now finishing its second year. This library provides building blocks for the development of numerical simulations for the solution of the fundamental equation of plasma physics: the Vlasov equation. In this context we have continued to add new modules improved interfaces and implemented 'continuous integration' software development techniques to improve code robustness and portability. Furthermore, we continue to involve other researchers within France and abroad to aid in the further development of this software product.

One of the aims of the ADT is to provide numerical building blocks for the GYSELA code developed at CEA Cadarache in collaboration with the Calvi project-team. GYSELA is used by physicists for simulating the development of turbulence in magnetic fusion plasmas in particular in view of the ITER project.

5.2. CLAC

Participants: Anaïs Crestetto, Philippe Helluy.

The objective of the three-dimensional parallel software CM2 (Code Multiéchelle Multiphysique) software is to implement a general solver for hyperbolic conservation laws. It is for instance able to solve the MHD model. CLAC is a C++ OpenCL/MPI based library derived from algorithms and ideas developed in CM2. CLAC means "Compute Language Approximation of Conservation laws".

It is clear now that a future supercomputer will be made of a collection of thousands of interconnected multicore processors. Globally it appears as a classical distributed memory MIMD machine. But at a lower level, each of the multicore processors is itself made of a shared memory MIMD unit (a few classical CPU cores) and a SIMD unit (a GPU). When designing new algorithms, it is important to adapt them for this architecture. Our philosophy will be to program our algorithms in such a way that they can be run efficiently on this kind of computers. Practically, we will use the MPI library for managing the high level parallelism, while the OpenCL library will efficiently operate the low level parallelism.

We have invested for several years now into scientific computing on GPU, using the open standard OpenCL (Open Computing Language). With Anaïs Crestetto, who is preparing a PhD in the CALVI project, we were recently awarded a prize in the international AMD OpenCL innovation challenge thanks. We have developed an OpenCL 2D Vlasov-Maxwell solver, coupling a PIC and a DG algorithms, which fully runs on a GPU. OpenCL is a very interesting tool because it is an open standard now available on almost all brands of multicore processors and GPU. The same parallel program can run on a GPU or a multicore processor without modification.

CLAC is written in C++, which is almost mandatory, because we use the OpenCL library. It also uses the MPI paradigm and is thus able to run on a cluster of GPU. CLAC is also inside a collaboration with a Strasbourg SME, AxesSim, which develops software for electromagnetic simulations. Thomas Strub, who is employed in AxesSim with a CIFRE position, is doing his Ph. D. on the conception and the development of CLAC applied to electromagnetic problems.

Because of the envisaged applications of CLAC, which may be either academical or commercial, it is necessary to conceive a modular framework. The heart of the library is made of generic parallel algorithms for solving conservation laws. The parallelism can be both fine grain (oriented towards GPU and multicore processors) and large grain (oriented towards GPU clusters). The separated modules allow managing the meshes and some specific applications. In this way, it is possible to isolate parts that can be protected by trade secret.

CORIDA Project-Team

5. Software

5.1. Simulation of viscous fluid-structure interactions

Participants: Takeo Takahashi [correspondant], Jean-François Scheid, Jérôme Lohéac.

A number of numerical codes for the simulation for fluids and fluid-structure problems has been developed by the team. These codes are mainly written in MATLAB Software with the use of C++ functions in order to improve the sparse array process of MATLAB. We have focused our attention on 3D simulations which require large CPU time resources as well as large memory storage. In order to solve the 3D Navier-Stokes equations which model the viscous fluid, we have implemented an efficient 3D Stokes sparse solver for MATLAB and a 3D characteristics method to deal with the nonlinearity of Navier-Stokes equations. This year, we have also started to unify our 2D fluid-structure codes (fluid alone, fluid with rigid bodies and fluid with fishes).

Another code has been developed in the case of self-propelled deformable object moving into viscous fluid. Our aim is to build a deformable ball which could swim in a viscous fluid. In order to do this we have started a collaboration with a team from the CRAN (Research Centre for Automatic Control). This software solves numerically 3D Stokes equations using finite elements methods. The source code is written for use with MATLAB thanks to a C++ library developped by ALICE.

- Version: v0.5
- Programming language: MATLABc++

5.2. Fish locomotion in perfect fluids with potential flow

Participants: Alexandre Munnier [correspondant], Marc Fuentes, Bruno Pinçon.

SOLEIL is a Matlab suite to simulate the self-propelled swimming motion of a single 3D swimmer immersed in a potential flow. The swimmer is modeled as a shape-changing body whose deformations can be either prescribed as a function of time (simulation of the direct swimming problem) or computed in such a way that the swimmer reaches a prescribed location (control problem). For given deformations, the hydrodynamical forces exerted by the fluid on the swimmer are expressed as solutions of 2D integral equations on the swimmer's surface, numerically solved by means of a collocation method.

SOLEIL is free, distributed under licence GPL v3. More details are available on the project web page http://soleil.gforge.inria.fr/.

The next step of SOLEIL (under progress) is to take into account a fluid whose flow is governed by Stokes equations.

- Version: 0.1
- Programming language:Matlab/C++

TOSCA Project-Team

5. Software

5.1. CarbonQuant

Participant: Mireille Bossy [correspondant].

CarbonQuant is a simulator project of CO2 allowances prices on a EU-ETS type market, by an indifference price approach.

It aims to demonstrate the high potentiality of stochastic control solvers, to quantify sensibilities of a carbon market with respect to its design.

Starting in September 2011, CarbonQuant is an ADT [1] Inria.

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

- Version: 0.1

[1] Technology Development Action

BIGS Project-Team

5. Software

5.1. Light diffusion into tissues

We are currently considering the possibility to implement our Matlab algorithms concerning light diffusion into tissues into the Matlab toolbox *Contsid*, developed by the System Identification team of the CRAN (http://www.iris.cran.uhp-nancy.fr/contsid/).

5.2. Online data analysis

A R package performing most of the methods of factorial analysis in an online way is under development by R. Bar and J-M. Monnez. Starting from a simulated data flow, the main goal of the program is to perform online factorial analyses (Principal Component Analyses, Canonical Correlation Analysis, Canonical Discriminant Analysis, Correspondence Analysis). Data are supposed to be independent and identically distributed observations of a random vector (whose distribution is a priori unknown). Defining stochastic approximation processes, the procedure is adaptative in the sense that the results of the analyses are updated recursively each time that a new data is taken into account.

From a theoretical point of view, the i.i.d case has been recently extended to the case of an expectation and/or covariance matrix of the random vector varying with time. We plan to include these improvements into our software.

5.3. Socio-economic index

A R package called SesIndexCreatoR has been written by B. Lalloué and J-M. Monnez in order to implement our socio-economic index for health inequalities. The version 1.0 of this package is currently freely available on the website of the Equit'Area project: http://www.equitarea.org/documents/packages_1.0-0/. It contains the functions needed to run the procedure (either integrally or partially) and obtain the corresponding SES index. The user may also create categories of this index with different methods (hierarchical clustering with or without k-nearest neighbors, quantiles, or intervals) and generate automatic reports of the results. Visualization and plotting functions are provided in the package.

CORTEX Project-Team

5. Software

5.1. Spiking neural networks simulation

Participants: Dominique Martinez, Yann Boniface.

A spiking neuron is usually modeled as a differential equation describing the evolution over time of its membrane potential. Each time the voltage reaches a given threshold, a spike is sent to other neurons depending on the connectivity. A spiking neural network is then described as a system of coupled differential equations. For the simulation of such a network we have written two simulation engines : (i) Mvaspike based on an event-driven approach and (ii) sirene based on a time-driven approach.

- Mvaspike : The event-driven simulation engine was developed in C++ and is available on http://mvaspike.gforge.inria.fr. Mvaspike is a general event-driven purpose tool aimed at modeling and simulating large, complex networks of biological neural networks. It allows to achieve good performance in the simulation phase while maintaining a high level of flexibility and programmability in the modeling phase. A large class of spiking neurons can be used ranging from standard leaky integrate-and-fire neurons to more abstract neurons, e.g. defined as complex finite state machines.
- Sirene : The time-driven simulator engine was written in C and is available on http://sirene.gforge.inria.fr. It has been developed for the simulation of biologically detailed models of neurons —such as conductance-based neurons— and synapses. Its high flexibility allows the user to implement easily any type of neuronal or synaptic model and use the appropriate numerical integration routine (e.g. Runge-Kutta at given order).

5.2. DANA: Implementation of computational neuroscience mechanisms

Participants: Nicolas Rougier, Mathieu Lefort, Wahiba Taouali.

Computational neuroscience is a vast domain of research going from the very precise modeling of a single spiking neuron, taking into account ion channels and/or dendrites spatial geometry up to the modeling of very large assemblies of simplified neurons that are able to give account of complex cognitive functions. DANA attempts to address this latter modeling activity by offering a Python computing framework for the design of very large assemblies of neurons using numerical and distributed computations. However, there does not exist something as a unified model of neuron: if the formal neuron has been established some sixty years ago, there exists today a myriad of different neuron models that can be used within an architecture. Some of them are very close to the original definition while some others tend to refine it by providing extra parameters or variables to the model in order to take into account the great variability of biological neurons. DANA makes the assumption that a neuron is essentially a set of numerical values that can vary over time due to the influence of other neurons and learning. DANA aims at providing a constrained and consistent Python framework that guarantee this definition to be enforced anywhere in the model, i.e., no symbol, no homonculus, no central executive.

5.3. ENAS: Event Neural Assembly Simulation

Participants: Frédéric Alexandre, Axel Hutt, Nicolas Rougier, Thierry Viéville.

EnaS (that stands for “Event Neural Assembly Simulation”) is a middleware implementing our last numerical and theoretical developments, allowing to simulate and analyze so called "event neural assemblies". The recent achievements include (in collaboration with the Neuromathcomp EPI): spike trains statistical analysis via Gibbs distributions, spiking network programing for exact event’s sequence restitution, discrete neural field parameters algorithmic adjustments and time-constrained event-based network simulation reconciling clock and event based simulation methods. It has been designed as plug-in for our simulators (e.g. DANA or Mvaspike) as other existing simulators (via the NeuralEnsemble meta-simulation platform) and additional modules for computations with neural unit assembly on standard platforms (e.g. Python or the Scilab platform).

5.4. OpenViBE

Participants: Laurent Bougrain, Octave Boussaton.

OpenViBE is a C++ open-source software devoted to the design, test and use of Brain-Computer Interfaces. The OpenViBE platform consists of a set of software modules that can be integrated easily and efficiently to design BCI applications. Key features of the platform are its modularity, its high-performance, its portability, its multiple-users facilities and its connection with high-end/Virtual Reality displays. The "designer" of the platform enables to build complete scenarios based on existing software modules using a dedicated graphical language and a simple Graphical User Interface (GUI). This software is available on the Inria Forge under the terms of the LGPL-V2 license.The development of OpenVibe is done in association with the Inria research team BUNRAKU for the national Inria project: ADT LOIC (*cf.* § 7.2).

5.5. CLONES: Closed-Loop Neural Simulations

Participant: Thomas Voegtlin.

The goal of this work is to provide an easy-to-use framework for closed-loop simulations, where interactions between the brain and body of an agent are simulated.

We developed an interface between the Sofa physics engine, (http://www.sofa-framework.org) and the Brian neural simulator (http://www.briansimulator.org). The interface consists in a Sofa plugin and a Python module for Brian. Sofa and Brian use different system processes, and communicate via shared memory. Synchronization between processes is achieved through semaphores.

As a demonstration of this interface, a physical model of undulatory locomotion in the nematode *c. elegans* was implemented, based on the PhD work of Jordan H. Boyle.

5.6. GINNet-DynNet: Decision-making platform

Participant: Marie Tonnelier.

GINNet (Graphical Interface for Neural Networks) is a decision-aid platform written in Java, intended to make neural network teaching, use and evaluation easier, by offering various parametrizations and several data pre-treatments. GINNet is based upon a local library for dynamic neural network developments called DynNet. DynNet (Dynamic Networks) is an object-oriented library, written in Java and containing base elements to build neural networks with dynamic architecture such as Optimal Cell Damage and Growing Neural Gas. Classical models are also already available (multi-layer Perceptron, Kohonen self-organizing maps, ...). Variable selection methods and aggregation methods (bagging, boosting, arcing) are implemented too.

The characteristics of GINNet are the following: Portable (100% Java), accessible (model creation in few clicks), complete platform (data importation and pre-treatments, parametrization of every models, result and performance visualization). The characteristics of DynNet are the following: Portable (100% Java), extensible (generic), independent from GINNet, persistent (results are saved in HML), rich (several models are already implemented), documented.

This platform is composed of several parts:

1. Data manipulation: Selection (variables, patterns), descriptive analysis (stat., PCA..), detection of missing, redundant data.
2. Corpus manipulation: Variable recoding, permutation, splitting (learning, validation, test sets).
3. Supervised networks: Simple and multi-layer perceptron.
4. Competitive networks: Kohonen maps, Neural Gas, Growing Neural Gas.
5. Metalearning: Arcing, bagging, boosting.
6. Results: Error curves, confusion matrix, confidence interval.

DynNet and GINNet are free softwares, registrated to the APP and distributed under CeCILL license, Java 1.4 compatible (http://ginnet.gforge.inria.fr). GINNet is available as an applet. For further information, see http://gforge.inria.fr/projects/ginnet (news, documentations, forums, bug tracking, feature requests, new releases...)

MASAIE Project-Team (section vide)

SHACRA Project-Team

5. Software

5.1. SOFA

SOFA, the Simulation Open Framework Architecture, is an international, multi-institution, collaborative initiative, aimed at developing a flexible and open source framework for interactive simulations. This will eventually establish new grounds for a widely usable standard system for long-term research and product prototyping, ultimately shared by many academic and industrial sites. Over the last two years, the SOFA framework has evolved from an informal collaborative work between the Sim Group at CIMIT, the Alcove, Asclepios and Evasion teams at Inria into a more structured development project. By proposing a unique architecture allowing the integration of the multiple competencies required for the development of a medical training system, we believe it will be possible to accelerate and foster research activities in the field of interactive medical simulation. The main objectives of the SOFA framework are:

- Simplify the development of medical simulation systems by improving interoperability
- Evaluate and validate new algorithms
- Accelerate the prototyping of simulation systems by promoting component reusability
- Promote collaboration between research groups
- Facilitate technology transfer between research and industry

Our activities around the SOFA framework will be twofold. We will remain one of the leading teams contributing to the design of SOFA, the development of its architecture and its distribution to research groups and industrial partners. In addition, we will use SOFA as a core element of most of our simulations, as a mean to facilitate the integration of results from partners of the national initiative, and to simplify the development of prototypes of simulation systems. For the past few years, there have been a few attempts at designing software toolkits for medical simulation. Examples include [36], GiPSi [25], SPORE [35] or SSTML [22]. These different solutions aim at the same goal: providing an answer (usually Open Source) to the various challenges of medical simulation research and development. Although our aim is similar, we propose a different approach, through a very modular and flexible software framework, while minimizing the impact of this flexibility on the computation overhead. To achieve these objectives, we have developed a new architecture that implements a series of innovative concepts. Also, by developing the SOFA framework collaboratively with scientific experts in the different areas of medical simulation, we believe we can provide state-of-the-art solutions that are generically applicable, yet computationally efficient. The following sections describe in more details our approach to the development of this framework, from a technical standpoint and from the perspective of a collaborative work.

5.1.1. SOFA architecture

Medical simulation relies on a variety of interacting physics-based models, such as rigid structures (e.g. bones), deformable structures (e.g. soft-tissues) and fluids. It also involves anatomical representations through geometrical models, used for visual rendering, collision detection or meshes that will support various computational models. Finally, interactions between these different models need to be efficient, accurate and capable of handling a variety of representations. In some instances, a hierarchy also exists between the various anatomical structures, and needs to be taken into account in the description of the simulated environment. The design of the SOFA architecture, by supporting these various requirements, brings the flexibility needed for academic research. Yet, its very efficient implementation makes it also suitable for professional applications and potentially for product development. This architecture relies on several innovative concepts, in particular the notion of multi-model representation. In SOFA, most simulation components (deformable models, collision models, medical devices, etc.) can have several representations, connected through a mechanism called mapping. Each representation is optimized for a particular task (e.g. collision detection, visualization) while at the same time

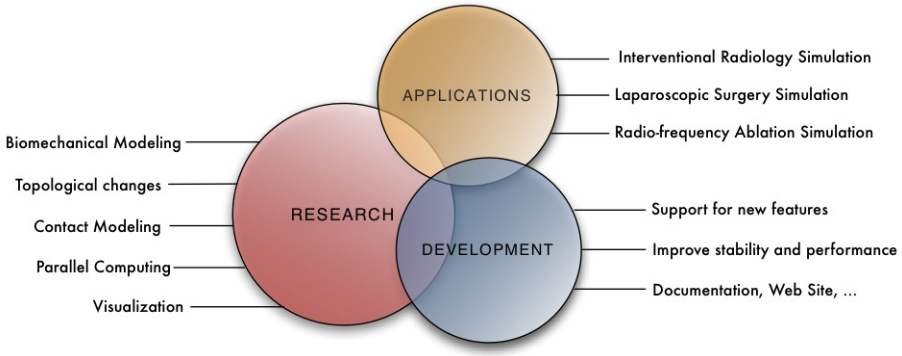

Figure 3. Multidisciplinary research and development of the SOFA framework need to take place simultaneously to quickly advance research in the field of computer-based interactive medical simulation

improving interoperability by creating a clear separation between the functional aspects of the simulation components. As a consequence, it is possible to have models of very different nature interact together, for instance rigid bodies, deformable objects, and fluids. This is an essential aspect of SOFA, as it will help the integration of new research components. This modular design also facilitates the rapid prototyping of simulation systems, allowing various combinations of algorithms to be tested and compared against each other. At a finer level of granularity, we also propose a decomposition of physical models (i.e. any model that behaves according to the laws of physics) into a set of basic components. In the case of (bio)mechanical models, which are computationally expensive, many strategies have been used to improve computation times or to reduce the complexity of the original model: linear elastic models have often been used instead of more complex non-linear representations, mass-spring methods as an alternative to finite element methods, etc. Each of these simplifications induces drawbacks, yet the importance of these drawbacks depends largely on the context in which they are applied. It becomes then very difficult to choose which particular method is most likely to provide the best results for a given simulation. To address this issue in SOFA we have introduced a finer level of granularity which permits to independently test and compare each component, such as time integration schemes, to see the change in performance or robustness of the simulation, or to test different constitutive models. These changes can be made in a matter of seconds, without having to recompile any of the code, by simply editing an XML file.

5.1.2. Current Results

Version 1.0 RC1 of SOFA was released in December 2011 but since October 2012, SOFA is now available through a public and anonymous SVN. More than 137,000 downloads of SOFA have been counted as of November 2012. More than 70 researchers, students, engineers have contributed at various degrees to SOFA, for a total of about 1,200,000 lines of code. Currently, thanks to its advanced architecture, SOFA allows to:

- Create complex and evolving simulations by combining new algorithms with existing algorithms
- Modify most parameters of the simulation by simply editing a XML file
- Build complex models from simpler ones using a scene-graph description
- Efficiently simulate the dynamics of interacting objects using abstract equation solvers
- Reuse and easily compare a variety of available methods
- Transparently parallelize complex computations using semantics based on data dependencies
- Use new generations of GPUs through the CUDA API to greatly improve computation times
- Use embedded Python environment to create interactive and parametric scenes, and interact with 3rd party software

Various results and information can be obtained on the SOFA website at http://www.sofa-framework.org. Most of the current results are generic and only aim at validating the different aspects of the SOFA framework. Developments of complex medical simulations have recently started, in particular in the areas of ophthalmic surgery and interventional radiology. We have also started a collaboration with a few companies (Digital Trainers, Didhaptics, B.K.) which are in the process of developing medical applications based on SOFA.

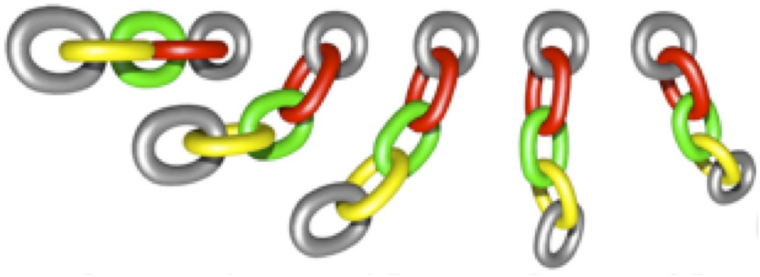

Figure 4. Animation of a chain combining a FEM model, a mass-spring model, a FFD grid, and a rigid body. This example is a perfect illustration of the flexibility of SOFA. Not only several algorithms for rigid or deformable bodies can be part of the same simulation, but they can also interact in a physically correct manner. No constraints between links were pre-defined, instead we relied on collision detection and stiff contact forces to handle the contacts. Using implicit integrator handling dynamically-created groups of interacting objects resulted in a stable simulation

ALGORILLE Project-Team

5. Software

5.1. Introduction

Software is a central part of our output. In the following we present the main tools to which we contribute. We use the Inria software self-assessment catalog for a classification.

5.2. parXXL

Participants: Jens Gustedt, Stéphane Vialle.

ParXXL is a library for large scale computation and communication that executes fine grained algorithms on coarse grained architectures (clusters, grids, mainframes). It is one of the software bases of the InterCell project and has been proven to be a stable support, there. It is available under a GPLv2 at http://parxxl.gforge.inria.fr/. ParXXL is not under active development anymore, but still maintained in the case of bugs or portability problems.
Software classification: A-3, SO-4, SM-3, EM-2, SDL-4, DA-4, CD-4, MS-2, TPM-2

5.3. Distem

Participants: Tomasz Buchert, Emmanuel Jeanvoine, Lucas Nussbaum, Luc Sarzyniec.

Wrekavoc and Distem are distributed system emulators. They enable researchers to evaluate unmodified distributed applications on heterogeneous distributed platforms created from an homogeneous cluster: CPU performance and network characteristics are altered by the emulator.
Wrekavoc was developed until 2010, and we then focused our efforts on **Distem**, that shares the same goals with a different design. Distem is available from http://distem.gforge.inria.fr/ under GPLv3.
Software classification: A-3-up, SO-4, SM-3-up, EM-3, SDL-4, DA-4, CD-4, MS-4, TPM-4.

5.4. SimGrid

Participants: Martin Quinson, Marion Guthmuller, Paul Bédaride, Lucas Nussbaum.

SimGrid is a toolkit for the simulation of distributed applications in heterogeneous distributed environments. The specific goal of the project is to facilitate research in the area of parallel and distributed large scale systems, such as Grids, P2P systems and clouds. Its use cases encompass heuristic evaluation, application prototyping or even real application development and tuning. SimGrid has an active user community of more than one hundred members, and is available under GPLv3 from http://simgrid.gforge.inria.fr/.
Software classification: A-4-up, SO-4, SM-4, EM-4, SDL-5, DA-4, CD-4, MS-3, TPM-4.

5.5. ORWL and P99

Participant: Jens Gustedt.

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

5.6. Kadeploy

Participants: Luc Sarzyniec, Emmanuel Jeanvoine, Lucas Nussbaum.

Kadeploy is a scalable, efficient and reliable deployment (provisioning) system for clusters and grids. It provides a set of tools for cloning, configuring (post installation) and managing cluster nodes. It can deploy a 300-nodes cluster in a few minutes, without intervention from the system administrator. It plays a key role on the Grid'5000 testbed, where it allows users to reconfigure the software environment on the nodes, and is also used on a dozen of production clusters both inside and outside INRIA. It is available from http://kadeploy3.gforge.inria.fr/ under the Cecill license.
Software classification: A-4-up, SO-3, SM-4, EM-4, SDL-4-up, DA-4, CD-4, MS-4, TPM-4.

MADYNES Project-Team

5. Software

5.1. SecSIP

Participants: Abdelkader Lahmadi [contact], Olivier Festor.

SecSip[1] is developed by the team to defend SIP-based (The Session Initiation Protocol) services from known vulnerabilities. It presents a proactive point of defense between a SIP-based network of devices (servers, proxies, user agents) and the open Internet. Therefore, all SIP traffic is inspected and analyzed against authored Veto specification before it is forwarded to these devices. When initializing, the SecSIP runtime starts loading and parsing authored VeTo blocks to identify different variables, event patterns, operations and actions from each rule. Veto is a generic declarative language for attack patterns specification. SecSIP implements an input and output layer, to capture, inject, send and receive SIP packets from and to the network. Intercepted packets are moved to the SIP Packet parser module. The main function of this module is to extract different fields within a SIP message and trigger events specified within the definition blocks. During each execution cycle when a SIP message arrives, the SecSIP runtime uses a data flow acyclic graph network to find definition matching rules and triggers defined events. The paired events in each operator node are propagated over the graph until a pattern is satisfied. When the pattern is satisfied, the respective rule is fired and the set of actions is executed.

SecSIP is freely available on the Internet. It was extended to support new protocols in the area of SCADA systems in 2012.

5.2. NDPMon

Participants: Isabelle Chrisment, Olivier Festor [contact].

The Neighbor Discovery Protocol Monitor (NDPMon) is an IPv6 implemention of the well-known ArpWatch tool. NDPMon monitors the pairing between IPv6 and Ethernet addresses (NDP activities: new station, changed Ethernet address, flip flop...). NDPMon also detects attacks on the NDP protocol, as defined in RFC 3756 (bogon, fake Router Advertisements...). New attacks based on the Neighbor Discovery Protocol and Address Auto-configuration (RFC 2461 and RFC 2462) have been identified and integrated in the tool. An XML file describes the default behavior of the network, with the authorized routers and prefixes, and a second XML document containing the neighbors database is used. This second file can be filled during a learning phase. All NDP activities are logged in the syslog utility, and so the attacks, but these ones are also reported by mail to the administrator. Finally, NDPMon can detect stack vulnerabilities, like the assignment of an Ethernet broadcast address on an interface.

NDPMon comes along with a WEB interface acting as a GUI to display the informations gathered by the tool, and give an overview of all alerts and reports. Thanks to color codes, the WEB interface makes possible for the administrator to have an history of what happened on his network and identify quickly problems. All the XML files used or produced by the daemon (neighbor cache, configuration file and alerts list) are translated in HTML via XSL for better readability. A statistic module is also integrated and gives informations about the discovery of the nodes and their type (MAC manufacturer distribution ...).

The software package and its source code is freely distributed under an opensource license (LGPL). It is implemented in C, and is available through a SourceForge project at http://ndpmon.sf.net. An open source community is now established for the tool which has distributions for several Operating Systems (Linux, FreeBSD, OpenBSD, NetBSD and Mac OS X). It is also integrated in FreeBSD ports at http://www.freebsd.org/cgi/cvsweb.cgi/ports/net-mgmt/ndpmon/. Binary distributions are also available for .deb and .rpm based Linux flavors.

In 2012, the software underwent a complete reshaping thanks to a substantial support from the High Security Lab which dedicated us 6 months of research engineer.

[1] http://secsip.gforge.inria.fr/doku.php

SCORE Team

5. Software

5.1. Wiki3.0

Participants: Luc André, Bogdan Flueras, Claudia-Lavinia Ignat [contact], Gérald Oster.

In the context of the Wiki 3.0 project (http://wiki30.xwikisas.com/) (december 2009 - june 2012) sponsored by the call for projects "Innovative Web" launched by the French Ministry of Economy, SCORE team designed and integrated real-time editing features into the XWiki system (http://www.xwiki.org). We designed solutions for a raw text editor as well as for a WYSIWYG editor for XWiki pages. The real-time wiki editor has been released as an extension of XWiki (http://extensions.xwiki.org/xwiki/bin/view/Extension/RealTime+Wiki+Editor).

ALICE Project-Team

4. Software

4.1. Graphite

Participants: Dobrina Boltcheva, Phuong Ho, Bruno Lévy, David Lopez, Romain Merland, Vincent Nivoliers, Jeanne Pellerin, Nicolas Ray.

Graphite is a research platform for computer graphics, 3D modeling and numerical geometry. It comprises all the main research results of our “geometry processing” group. Data structures for cellular complexes, parameterization, multi-resolution analysis and numerical optimization are the main features of the software. Graphite is publicly available since October 2003. It is hosted by Inria GForge since September 2008 (1000 downloads in two months). Graphite is one of the common software platforms used in the frame of the European Network of Excellence AIMShape .

4.2. MicroMegas

Participant: Samuel Hornus.

Micromegas is a 3D modeler, developed as a plugin of Graphite, dedicated to molecular biology. It is developed in cooperation with the Fourmentin Guilbert foundation and has recently been renamed "GraphiteLife-Explorer". Biologists need simple spatial modeling tools to help in understanding the role of objects’ relative position in the functioning of the cell. In this context, we develop a tool for easy DNA modeling. The tool generates DNA along any user-given curve, open or closed, allows fine-tuning of atoms’ position and, most importantly, exports to PDB.

In 2012, its development has been actively pursued by Samuel Hornus in the first trimester. The software is freely downloadable. A paper describing will appear in the broad journal PLOS One [9]. A poster was also presented at the European Conference on Computational Biology in september 2012.

4.3. CGAL package for Delaunay triangulations

Participant: Samuel Hornus.

This year was devolved also to finishing touches on the CGAL package for Delaunay triangulations (3rd submission to the CGAL editorial board).

Following the reviews for the second submission, Samuel Hornus has collaborated with Olivier Devillers (Inria Sophia Antipolis) to put the finishing touches to a new CGAL package for Delaunay triangulation in any dimension. It provides exact construction of Delaunay triangulations, supporting both the addition and deletion of vertices. The code takes the form of a collection of C++ template classes to ensures high performance when specializing the code to a given euclidian dimension.

4.4. OpenNL - Open Numerical Library

Participants: Thomas Jost, Bruno Lévy, Nicolas Ray, Rhaleb Zayer.

OpenNL is a standalone library for numerical optimization, especially well-suited to mesh processing. The API is inspired by the graphics API OpenGL, this makes the learning curve easy for computer graphics practitioners. The included demo program implements our LSCM [5] mesh unwrapping method. It was integrated in Blender by Brecht Van Lommel and others to create automatic texture mapping methods. OpenNL is extended with two specialized modules :

- CGAL parameterization package: this software library, developed in cooperation with Pierre Alliez and Laurent Saboret, is a CGALpackage for mesh parameterization.
- Concurrent Number Cruncher: this software library extends OpenNL with parallel computing on the GPU, implemented using the CUDA API.

4.5. Intersurf

Participants: Xavier Cavin, Nicolas Ray.

Intersurf is a plugin of the VMD (Visual Molecular Dynamics) software. VMD is developed by the Theoretical and Computational Biophysics Group at the Beckmann Institute at University of Illinois. The Intersurf plugin is released with the official version of VMD since the 1.8.3 release. It provides surfaces representing the interaction between two groups of atoms, and colors can be added to represent interaction forces between these groups of atoms. We plan to include in this package the new results obtained this year in molecular surface visualization by Matthieu Chavent.

4.6. LibSL

Participants: Anass Lasram, Sylvain Lefebvre.

LibSL is a Simple library for graphics. Sylvain Lefebvre continued development of the LibSL graphics library (under CeCill-C licence, filed at the APP). LibSL is a toolbox for rapid prototyping of computer graphics algorithms, under both OpenGL, DirectX 9/10, Windows and Linux. The library is actively used in both the REVES / Inria Sophia-Antipolis and the Alice / Inria Nancy Grand-Est teams.

MAGRIT Project-Team

5. Software

5.1. Software

Our software efforts are integrated in a library called RAlib which contains our research development on image processing, registration (2D and 3D) and visualization. This library is licensed by the APP (French agency for software protection).

The visualization module is called QGLSG: it enables the visualization of images, 2D and 3D objects under a consistent perspective projection. It is based on Qt [1] and OpenScenegraph [2] libraries. The QGLSG library integrates innovative features such as online camera distortion correction, and invisible objects that can be incorporated in a scene so that virtual objects can cast shadows on real objects, and occlusion between virtual and real objects are easier to handle. The library was also ported to Mac OS and Windows and a full doxygen documentation was written.

[1] http://qt.digia.com
[2] http://www.openscenegraph.org/projects/osg

MAIA Project-Team

5. Software

5.1. AA4MM

Participants: Vincent Chevrier [correspondant], Benjamin Camus.

Laurent Ciarletta (Madynes team, LORIA) is a collaborator and correspondant for this software.

AA4MM (Agents and Artefacts for Multi-modeling and Multi-simulation) is a framework for coupling existing and heterogeneous models and simulators in order to model and simulate complex systems. The first implementation of the AA4MM meta-model was proposed in Julien Siebert's PhD [56] and written in Java. This year we added a new coupling between models to represent multi-level modeling, and rewrote a part of the core to ease coupling of simulator.

5.2. MASDYNE

Participant: Vincent Chevrier [correspondant].

This work was undertaken in a joint PhD Thesis between MAIA and Madynes Team. Laurent Ciarletta (Madynes team, LORIA) has been co-advisor of this PhD and correspondant for this software.

Other contributors to this software were: Tom Leclerc, François Klein, Christophe Torin, Marcel Lamenu, Guillaume Favre and Amir Toly.

MASDYNE (Multi-Agent Simulator of DYnamic Networks usErs) is a multi-agent simulator for modeling and simulating users behaviors in mobile ad hoc network. This software is part of joint work with MADYNES team, on modeling and simulation of ubiquitous networks.

5.3. FiatLux

FiatLux is a discrete dynamical systems simulator that allows the user to experiment with various models and to perturb them. Its main feature is to allow users to change the type of updating, for example from a deterministic parallel updating to an asynchronous random updating. FiatLux has a Graphical User Interface and can also be launched in a batch mode for the experiments that require statistics. In 2012, the main contributions were made by Olivier Bouré, who developed a lattice-gas cellular automata module.

5.4. Cart-o-matic

Participants: Olivier Simonin [correspondant], François Charpillet, Antoine Bautin, Nicolas Beaufort.

Philippe Lucidarme (Université d'Angers, LISA) is a collaborator and the coordinator of the Cartomatic project.

Cart-o-matic is a software platform for (multi-)robot exploration and mapping tasks. It has been developed by Maia members and LISA (Univ. Angers) members during the robotics ANR/DGA Carotte challenge (2009-2012). This platform is composed of three softwares which as been protected by software copyrights (APP): Slam-o-matic a SLAM algorithm developed by LISA members, Plan-o-matic a robot trajectory planning algorithm developed by Maia and LISA members and Expl-o-matic a distributed multi-agent strategy for multi-robot exploration developed by Maia members (which is based on algorithms proposed in the PhD Thesis of Antoine Bautin). Cf. illustration at Cart-o-matic

The purchase of Cart-o-matic by some robotics companies is underway.

ORPAILLEUR Project-Team

5. Software

5.1. Generic Symbolic KDD Systems

5.1.1. The Coron Platform

Participants: Victor Codocedo, Adrien Coulet, Amedeo Napoli, Yannick Toussaint, Jérémie Bourseau [contact person].

data mining, frequent itemsets, frequent closed itemsets, frequent generators, association rule extraction, rare itemsets

The Coron platform [120], [102] is a KDD toolkit organized around three main components: (1) Coron-base, (2) AssRuleX, and (3) pre- and post-processing modules. The software was registered at the "Agence pour la Protection des Programmes" (APP) and is freely available (see http://coron.loria.fr). The Coron-base component includes a complete collection of data mining algorithms for extracting itemsets such as frequent itemsets, frequent closed itemsets, frequent generators. In this collection we can find APriori, Close, Pascal, Eclat, Charm, and, as well, original algorithms such as ZART, Snow, Touch, and Talky-G. The Coron-base component contains also algorithms for extracting rare itemsets and rare association rules, e.g. APriori-rare, MRG-EXP, ARIMA, and BTB. AssRuleX generates different sets of association rules (from itemsets), such as minimal non-redundant association rules, generic basis, and informative basis. In addition, the Coron system supports the whole life-cycle of a data mining task and proposes modules for cleaning the input dataset, and for reducing its size if necessary. The Coron toolkit is developed in Java, is operational, and was already used in several research projects.

5.1.2. Orion: Skycube Computation Software

Participant: Chedy Raïssi [contact person].

skyline, skycube algorithms

This program implements the algorithms described in a research paper published last year at VLDB 2010 [112]. The software provides a list of four algorithms discussed in the paper in order to compute skycubes. This is the most efficient –in term of space usage and runtime– implementation for skycube computation (see https://github.com/leander256/Orion).

5.2. Stochastic systems for knowledge discovery and simulation

5.2.1. The CarottAge system

Participants: Florence Le Ber, Jean-François Mari [contact person].

Hidden Markov Models, stochastic process

The system CarottAge is based on Hidden Markov Models of second order and provides a non supervised temporal clustering algorithm for data mining. It is freely available under GPL license (see http://www.loria.fr/~jfmari/App/).

It provides a synthetic representation of temporal and spatial data. CarottAge is currently used by INRA researchers interested in mining the changes in territories related to the loss of biodiversity (projects ANR BiodivAgrim and ACI Ecoger) and/or water contamination. A new version incorporating a graphic user interface was released and is now running on Windows systems.

CarottAge has been used for mining hydromorphological data. Actually a comparison was performed with three other algorithms classically used for the delineation of river continuum and CarottAge proved to give very interesting results for that purpose [17].

5.2.2. The ARPEnTAge system

Participants: Florence Le Ber, Jean-François Mari [contact person].

Hidden Markov Models, stochastic process

ARPEnTAge [1] (for *Analyse de Régularités dans les Paysages: Environnement, Territoires, Agronomie* is a software based on stochastic models (HMM2 and Markov Field) for analyzing spatio-temporal data-bases [106]. ARPEnTAge is built on top of the CarottAge system to fully take into account the spatial dimension of input sequences. It takes as input an array of discrete data in which the columns contain the annual land-uses and the rows are regularly spaced locations of the studied landscape. It performs a Time-Space clustering of a landscape based on its time dynamic Land Uses (LUS). Displaying tools and the generation of Time-dominant shape files have also been defined.

We model the spatial structure of the landscape by a Potts model with external field whose sites are LUS located in the parcels. The dynamics of these LUS are modeled by a temporal HMM2. This leads to the definition of a Potts model where the underlying mean field is approximated by a hierarchical hidden Markov model that processes a Hilbert-Peano fractal curve spanning the image.

Those stochastic models have been used to segment the landscape into patches, each of them being characterized by a temporal HMM2. The patch labels, together with the geographic coordinates, determine a clustered image of the landscape that can be coded within an ESRI shapefile. ARPEnTAge can locate in a 2-D territory time regularities and implements a Time-dominant approach in Geographic Information Systems.

ARPEnTAge is freely available (GPL license) and is currently used by INRA researchers interested in mining the changes in territories related to the loss of biodiversity (projects ANR BiodivAgrim and ACI Ecoger) and/or water contamination.

In these practical applications, CarottAge and ARPEnTAge aim at building a partition –called the hidden partition– in which the inherent noise of the data is withdrawn as much as possible. The estimation of the model parameters is performed by training algorithms based on the Expectation Maximization and Mean Field theories. The ARPEnTAge system takes into account: (i) the various shapes of the territories that are not represented by square matrices of pixels, (ii) the use of pixels of different size with composite attributes representing the agricultural pieces and their attributes, (iii) the irregular neighborhood relation between those pixels, (iv) the use of shape files to facilitate the interaction with GIS (geographical information system).

ARPEnTAge and CarottAge have been used for mining decision rules in a territory holding environmental issues. They provide a way of visualizing the impact of farmers decision rules in the landscape and revealing new extra hidden decision rules [23].

5.2.3. GenExp-LandSiTes: KDD and simulation

Participants: Sébastien Da Silva, Florence Le Ber [contact person], Jean-François Mari.

simulation, Hidden Markov Models

In the framework of the project “Impact des OGM” initiated by the French ministry of research, we have developed a software called GenExp-LandSiTes for simulating bidimensional random landscapes, and then studying the dissemination of vegetable transgenes. The GenExp-LandSiTes system is linked to the CarottAge system, and is based on computational geometry and spatial statistics. The simulated landscapes are given as input for programs such as “Mapod-Maïs” or “GeneSys-Colza” for studying the transgene diffusion. Other landscape models based on tessellation methods are under studies. The last version of GenExp allows an interaction with R and deals with several geographical data formats.

[1] http://www.loria.fr/~jfmari/App/

This work is now part of an INRA research network about landscape modeling, PAYOTE, that gathers several research teams of agronomists, ecologists, statisticians, and computer scientists. Sébastien da Silva is preparing his PhD thesis within this framework and is conducted both by Claire Lavigne (DR in ecology, INRA Avignon) and Florence Le Ber [46], [40].

GenExp-LandSiTes was part of a survey about innovative tools for geographical information [74], [73]. This survey has been conducted within the GDR Magis and has been presented in a book both in French and in English.

5.3. KDD in Systems Biology

5.3.1. IntelliGO online

The IntelliGO measure computes semantic similarity between terms from a structured vocabulary (Gene Ontology: GO) and uses these values for computing functional similarity between genes annotated by sets of GO terms [83]. The IntelliGO measure is made available on line (http://plateforme-mbi.loria.fr/intelligo/) to be used by members of the community for exploitation and evaluation purposes. It is possible to compute the functional similarity between two genes, the intra-set similarity value in a given set of genes, and the inter-set similarity value for two given sets of genes.

5.3.2. WAFOBI : KNIME nodes for relational mining of biological data

KNIME (for "Konstanz Information Miner") is an open-source visual programming environment for data integration, processing, and analysis. KNIME has been developed using rigorous software engineering practices and is used by professionals in both industry and academia. The KNIME environment includes a rich library of data manipulation tools (import, export) and several mining algorithms which operate on a single data matrix (decision trees, clustering, frequent itemsets, association rules...). The KNIME platform aims at facilitating the data mining experiment settings as many tests are required for tuning the mining algorithms. The evaluation of the mining results is also an important issue and its configuration is made easier.

A position of engineer ("Ingénieur Jeune Diplomé Inria") was granted to the Orpailleur team to develop some extra KNIME nodes for relational data mining using the ALEPH program (http://www.comlab.ox.ac.uk/oucl/research/areas/machlearn/Aleph/aleph.pl). The developed KNIME nodes include a data preparation node for defining a set of first-order predicates from a set of relation schemes and then a set of facts from the corresponding data tables (learning set). A specific node allows to configure and run the ALEPH program to build a set of rules. Subsequent nodes allow to test the first-order rules on a test set and to perform configurable cross validations. An Inria APP procedure is currently pending.

5.3.3. MOdel-driven Data Integration for Mining (MODIM)

Participants: Marie-Dominique Devignes [contact person], Malika Smaïl-Tabbone.

The MODIM software (MOdel-driven Data Integration for Mining) is a user-friendly data integration tool which can be summarized along three functions: (i) building a data model taking into account mining requirements and existing resources; (ii) specifying a workflow for collecting data, leading to the specification of wrappers for populating a target database; (iii) defining views on the data model for identified mining scenarios. A steady-version of the software has been deposited through Inria APP procedure in December, 2010.

Although MODIM is domain independent, it was used so far for biological data integration in various internal research studies. A poster was presented at the last JOBIM conference (Paris, June 2011). Recently, MODIM was used by colleagues from the LIFL for organizing data about non ribosomal peptide syntheses. Feedback from users led to extensions of the software. The sources can be downloaded at https://gforge.inria.fr/projects/modim/.

5.4. Knowledge-Based Systems and Semantic Web Systems

5.4.1. The Kasimir System for Decision Knowledge Management

Participants: Nicolas Jay, Jean Lieber [contact person], Amedeo Napoli, Thomas Meilender.

classification-based reasoning, case-based reasoning, edition and maintenance of knowledge, decision knowledge management, semantic portal

The objective of the Kasimir system is decision support and knowledge management for the treatment of cancer. A number of modules have been developed within the Kasimir system for editing of treatment protocols, visualization, and maintenance. Kasimir is developed within a semantic portal, based on OWL. KatexOWL (Kasimir Toolkit for Exploiting OWL Ontologies, http://katexowl.loria.fr) has been developed in a generic way and is applied to Kasimir. In particular, the user interface EdHibou of KatexOWL is used for querying the protocols represented within the Kasimir system.

The software CabamakA (case base mining for adaptation knowledge acquisition) is a module of the Kasimir system. This system performs case base mining for adaptation knowledge acquisition and provides information units to be used for building adaptation rules. Actually, the mining process in CabamakA is implemented thanks to a frequent close itemset extraction module of the Coron platform (see §5.1.1).

The Oncologik system is a collaborative editing tool aiming at facilitating the management of medical guidelines [49], [48]. Based on a semantic wiki, it allows the acquisition of formalized decision knowledge. A production version was released this year (http://www.oncologik.fr/). Oncologik also includes a graphical decision tree editor, KcatoS [61].

5.4.2. Taaable: a system for retrieving and creating new cooking recipes by adaptation

Participants: Valmi Dufour-Lussier, Emmanuelle Gaillard, Laura Infante Blanco, Florence Le Ber, Jean Lieber, Amedeo Napoli, Emmanuel Nauer [contact person].

knowledge acquisition, ontology engineering, semantic annotation, case-based reasoning, hierarchical classification, text mining

Taaable is a system whose objectives are to retrieve textual cooking recipes and to adapt these retrieved recipes whenever needed. Suppose that someone is looking for a “leek pie” but has only an “onion pie” recipe: how can the onion pie recipe be adapted?

The Taaable system combines principles, methods, and technologies of knowledge engineering, namely case-based reasoning (CBR), ontology engineering, text mining, text annotation, knowledge representation, and hierarchical classification. Ontologies for representing knowledge about the cooking domain, and a terminological base for binding texts and ontology concepts, have been built from textual web resources. These resources are used by an annotation process for building a formal representation of textual recipes. A CBR engine considers each recipe as a case, and uses domain knowledge for reasoning, especially for adapting an existing recipe w.r.t. constraints provided by the user, holding on ingredients and dish types.

The Taaable system is available since 2008 on line at http://taaable.fr, but is constantly evolving. This year, Taaable has been extended by two new features, both concerning knowledge acquisition.

The first feature uses closed itemsets for extracting adaptation knowledge in order to better adapt recipes. A first approach integrates a previous work about adaptation rule extraction [93] into a collaborative environment, in which humans and machines may now collaborate to better acquire adaptation rules [38]. This environment integrates also the results of a new work on knowledge extraction where specific cooking adaptation rules that can be applied to a single recipe, are generalized using close itemsets into generic adaptation rules, to make them usable on other recipes [60].

The second feature addresses the improvement of the formal representation of the preparation part of recipes, using a semi-automatic annotation process [59]. In Taaable, the procedural text describing the preparation is formalized in a graph, where cooking actions and ingredients, among others, are represented as vertexes, and semantic relations between those, shown as arcs. As the automatic annotation process that transforms, using natural language processing, a procedural text into a graph, produces incomplete annotation (disconnected graphs) or other annotation errors, a validating and correcting step is required. A specific graphical interface has been built to provide the users with a way to correct the graph representation of the cooking process, improving at the same time the quality of the knowledge about cooking procedures.

5.4.3. Tuuurbine: a generic ontology guided case-based inference engine

Participants: Laura Infante Blanco, Jean Lieber, Emmanuel Nauer [contact person].

case-based reasoning, inference engine, knowledge representation, ontology engineering, semantic web

The experience acquired since 5 years with the Taaable system conducted to the creation of a generic cased-based reasoning system, whose reasoning procedure is based on a domain ontology. This new system, called Tuuurbine, takes into account the retrieval step, the case base organization, but also an adaptation procedure which is not addressed by other generic case-based reasoning tools. Moreover, Tuuurbine is built over semantic web standards that will ensure facilities for being plugged over data available on the web. The domain knowledge is considered to be represented in a RDF store, which could be additionally be interfaced with a semantic wiki, in order to benefit from the collaborative edition and management of the knowledge involved in the reasoning system (cases, ontology, adaptation rules). This development is support by an Inria ADT funding.

PAROLE Project-Team

5. Software

5.1. WinSnoori

contact : Yves Laprie (Yves.Laprie@loria.fr)

WinSnoori is a speech analysis software that we have been developing for 15 years. It is intended to facilitate the work of the scientist in automatic speech recognition, phonetics or speech signal processing. Basic functions of Snoori enable several types of spectrograms to be calculated and the fine edition of speech signals (cut, paste, and a number of filters) as the spectrogram allows the acoustical consequences of all the modifications to be evaluated. Beside this set of basic functions, there are various functionalities to annotate phonetically or orthographically speech files, to extract fundamental frequency, to pilot the Klatt synthesizer and to utilize PSOLA resynthesis.

The main improvement concerns automatic formant tracking which is now available with other tools for copy synthesis. It is now possible to determine parameters for the formant synthesizer of Klatt quite automatically. The first step is formant tracking, then the determination of F0 parameters and finally the adjustment of formant amplitudes for the parallel branch of the Klatt synthesizer enable a synthetic speech signal to be generated. The automatic formant tracking that has been implemented is an improved version of the concurrent curve formant tracking [49]. One key point of this tracking algorithm is the construction of initial rough estimates of formant trajectories. The previous algorithm used a mobile average applied onto LPC roots. The window is sufficiently large (200 ms) to remove fast varying variations due to the detection of spurious roots. The counterpart of this long duration is that the mobile average prevents formants fairly far from the mobile average to be kept. This is particularly sensitive in the case of F2 which presents low frequency values for back vowels. A simple algorithm to detect back vowels from the overall spectral shape and particularly energy levels has been added in order to keep extreme values of F2 which are relevant.

Together with other improvements reported during the last years, formant tracking enables copy synthesis. The current version of WinSnoori is available on http://www.winsnoori.fr.

5.2. JSnoori

contact : Yves Laprie (Yves.Laprie@loria.fr)

JSnoori is written in Java and uses signal processing algorithms developed within WinSnoori software with the double objective of being a platform independent signal visualization and manipulation tool, and also for designing exercises for learning the prosody of a foreign language. JSnoori thus focused the calculation of F0, the forced alignment of non native English uttered by French speakers and the correction of prosody parameters (F0, rhythm and energy). Since phonetic segmentations and annotations play a central role in the derivation of diagnosis concerning the realization of prosody by learners, several tools have been incorporated to segment and annotate speech. In particular, a complete phonetic keyboard is available, several kinds of annotation can be used (phonemes, syllables and words) and forced alignment can exploit variants to cope with non native accents. In addition, JSnoori offers real time F0 calculation which can be useful from a pedagogical point of view.

5.3. Xarticulator

contact : Yves Laprie (Yves.Laprie@loria.fr)

Xarticulators software is intended to delineate contours of speech articulators in X-ray images, construct articulatory models and synthesize speech from X-ray films. This software provide tools to track contours automatically, semi-automatically or by hand, to make the visibility of contours easier, to add anatomical landmarks to speech articulators and to synchronize images together with the sound.

It also enables the construction of adaptable linear articulatory models from the X-ray images.

This year we particularly worked on the possibility of synthesizing speech from X-ray images. We thus designed an algorithm to compute the centerline of the vocal tract in order to segment the vocal tract into elementary tubes approximating the propagation of a one-dimensional wave. In addition we also added the possibility of processing digitized manual delineation results made on sheet of papers when no software was available

5.4. SUBWEB

contacts : David Langlois (langlois@loria.fr) and Kamel Smaïli (smaili@loria.fr).

We published in 2007 a method which allows to align sub-titles comparable corpora [50]. In 2009, we proposed an alignment web tool based on the developed algorithm. It allows to: upload a source and a target files, obtain an alignment at a sub-title level with a verbose option, and and a graphical representation of the course of the algorithm. This work has been supported by CPER/TALC/SUBWEB [2].

5.5. SELORIA

contact : Odile Mella (Odile.Mella@loria.fr).

SELORIA is a toolbox for speaker diarization.

The system contains the following steps:

- Speaker change detection: to find points in the audio stream which are candidates for speaker change points, a distance is computed between two Gaussian modeling data of two adjacent given-length windows. By sliding both windows on the whole audio stream, a distance curve is obtained. A peak in this curve is thus considered as a speaker change point.
- Segment recombination: too many speaker turn points detected during the previous step results in a lot of false alarms. A segment recombination using BIC is needed to recombine adjacent segments uttered by the same speaker.
- Speaker clustering: in this step, speech segments of the same speaker are clustered. Top-down clustering techniques or bottom-up hierarchical clustering techniques using BIC can be used.
- Viterbi re-segmentation: the previous clustering step provides enough data for every speaker to estimate multi-gaussian speaker models. These models are used by a Viterbi algorithm to refine the boundaries between speakers.
- Second speaker clustering step (called cluster recombination): This step uses Universal Background Models (UBM) and the Normalized Cross Likelihood Ratio (NCLR) measure.

This toolbox is derived from mClust designed by LIUM.

5.6. ANTS

contacts : Dominique Fohr (fohr@loria.fr) and Denis Jouvet (denis.jouvet@inria.fr).

The aim of the Automatic News Transcription System (ANTS) is to transcribe radio or TV shows. ANTS is composed of several stages. The first processing steps aim at splitting the audio stream into homogeneous segments of a manageable size and at identifying the segment characteristics in order to allow the use of specific algorithms or models according to the nature of the segment. This includes broad-band/narrow-band speech segmentation, speech/music classification, speaker segmentation and clustering, detection of silences/breathing segments and generally speaker gender classification.

[2] http://wikitalc.loria.fr/dokuwiki/doku.php?id=operations:subweb

Each segment is then decoded using a large vocabulary continuous speech recognition engine, either the Julius engine or the Sphinx engine. The Julius engine operates in two passes: in the first pass, a frame-synchronous beam search algorithm is applied on a tree-structured lexicon assigned with bigram language model probabilities. The output of this pass is a word-lattice. In the second pass, a stack decoding algorithm using a trigram language model gives the N-best recognition sentences. The Sphinx engine processes the speech input segment in a single forward pass using a trigram language model.

Further processing passes are usually run in order to apply unsupervised adaptation processes on the feature computations (VTLN: vocal tract length normalization) and/or on the model parameters (MLLR: maximum likelihood linear regression), or to use speaker adaptive training (SAT) based models. Moreover decoding results of both systems can be efficiently combined for improved decoding performance.

The latest version which relies on a perl script exploits the multiple CPUs available on a computer to reduce the processing time, and runs on both a stand alone linux machine and on the cluster.

5.7. JTrans

Contact : Christophe Cerisara (Christophe.Cerisara@loria.fr).

JTrans is an open-source software for semi-automatic alignement of speech and textual corpus. It is written 100% in JAVA and exploits libraries developed since several years in our team. Two algorithms are available for automatic alignment: a block-viterbi and standard forced-alignement Viterbi. The latter is used when manual anchors are defined, while the former is used for long audio files that do not fit in memory. It is designed to be intuitive and easy to use, with a focus on GUI design. The rationale behind JTrans is to let the user control and check on-the-fly the automatic alignment algorithms. It is bundled for now with a French phonetic lexicon and French models.

Recent improvements include its integration within the JSafran platform and its release as a Java applet that can be demonstrated on web pages. During the last three months, JTrans has been downloaded about 120 times and seven users of JTrans, outside LORIA, have directly contacted the team for requests about JTrans.

JTrans is developed in the context of the CPER MISN TALC project, in collaboration between the Parole and Talaris Inria teams, and CNRS researchers from the ATILF laboratory. It is distributed under the Cecill-C licence, and can be downloaded at http://synalp.loria.fr/?n=Research.Software

5.8. CoALT

contacts : Dominique Fohr (dominique.fohr@loria.fr) and Odile Mella (odile.mella@loria.fr).

CoALT (Comparing Automatic Labelling Tools) compares two automatic labellers or two speech-text alignment tools, ranks them and displays statistics about their differences. The main feature of our software is that a user can define its own criteria for evaluating and comparing two speech- text alignment tools. With CoALT, a user can give more importance to either phoneme labels or phoneme boundaries because the CoALT elastic comparison algorithm takes into account time boundaries. Moreover, by providing a set of phonetic rules, a user can define the allowed discrepancies between the automatic labelling result and the hand-labelling one.

5.9. TTS SoJA

contact : Vincent Colotte (Vincent.Colotte@loria.fr).

TTS SoJA (Speech synthesis platform in Java) is a software of text-to-speech synthesis system. The aim of this software is to provide a toolkit to test some steps of natural language processing and to provide a whole system of TTS based on non uniform unit selection algorithm. The software performs all steps from text to the speech signal. Moreover, it provides a set of tools to elaborate a corpus for a TTS system (transcription alignment, ...). Currently, the corpus contains 1800 sentences (about 3 hours of speech) recorded by a female speaker.

Most of the modules are developed in Java. Some modules are in C. The platform is designed to make easy the addition of new modules. The software runs under Windows and Linux (tested on Mandriva, Ubuntu). It can be launch with a graphical user interface or directly integrated in a Java code or by following the client-server paradigm.

The software license should easily allow associations of impaired people to use the software. A demo web site has been built: http://soja-tts.loria.fr

5.10. Corpus Recorder

contact : Vincent Colotte (Vincent.Colotte@loria.fr).

Corpus Recorder is a software for the recording of audio corpora. It provides a easy tool to record with a microphone. The gain of the audio input is controlled during the recording. From a list of sentences, the output is a set of wav files automatically renamed with textual information given in input (nationality, speaker language, gender...). An easy syntactic tagging allows to display a textual context of the sentence to pronounce. This software is suitable for recording sentences with information to guide the speaker.

The software is developed in Tcl/Tk (tested under Windows and Linux). It was used for the recording of sentences for the TTS system SOJA and during the Intonale Project (Prosody Modeling).

5.11. VisArtico

contact : Slim Ouni (Slim.Ouni@loria.fr).

VisArtico is intended to visualize articulatory data acquired using an articulograph [30], [29]. It is intended for researchers that need to visualize data acquired from the articulograph with no excessive processing. It is well adapted to the data acquired using the AG500 and AG501 (developed by Carstens Medizinelektronik GmbH), and the articulograph NDI Wave, developed by Northern Digital Inc.

The software allows displaying the positions of the sensors that are simultaneously animated with the speech signal. It is possible to display the tongue contour and the lips contour. The software helps to find the midsagittal plane of the speaker and find the palate contour. In addition, VisArtico allows labeling phonetically the articulatory data.

All this information is very useful to researchers working in the field of speech production, as phoneticians for instance. VisArtico provides several possible views: (1) temporal view, (2) 3D spatial view and (3) 2D midsagittal view. In the temporal view, it is possible to display different articulatory trajectories in addition to the acoustic signal and eventually labels. The midsagittal view can display the tongue contour, the jaw, the lips and the palate.

VisArtico provides several tools to help to improve the quality of interpreting the data. It is cross-platform software as it is developed in JAVA and does not need any external library or framework to be additionally installed. It was tested and worked on Windows, Mac OS, and Linux. It should work on any system having JAVA installed. VisArtico is freely distributed via a dedicated website http://visartico.loria.fr.

SEMAGRAMME Team

5. Software

5.1. Leopar

Participants: Bruno Guillaume [correspondant], Guy Perrier.

Interaction Grammar, parsing

5.1.1. Software description

Leopar is a parser for natural languages which is based on the formalism of Interaction Grammars [30]. It uses a parsing principle, called "electrostatic parsing" which consists in neutralizing opposite polarities. A positive polarity corresponds to an available linguistic feature and a negative one to an expected feature.

Parsing a sentence with an Interaction Grammar consists in first selecting a lexical entry for each of its words. A lexical entry is an underspecified syntactic tree, a tree description in other words. Then, all selected tree descriptions are combined by partial superposition guided by the aim of neutralizing polarities: two opposite polarities are neutralized by merging their support nodes. Parsing succeeds if the process ends with a minimal and neutral tree. As IGs are based on polarities and under-specified trees, Leopar uses some specific and non-trivial data-structures and algorithms.

The electrostatic principle has been intensively considered in Leopar. The theoretical problem of parsing IGs is NP-complete; the nondeterminism usually associated to NP-completeness is present at two levels: when a description for each word is selected from the lexicon, and when a choice of which nodes to merge is made. Polarities have shown their efficiency in pruning the search tree:

- In the first step (tagging the words of the sentence with tree descriptions), we forget the structure of descriptions, and only keep the bag of their features. In this case, parsing inside the formalism is greatly simplified because composition rules reduce to the neutralization of a negative feature-value pair $f \longleftarrow v$ by a dual positive feature-value pair $f \longrightarrow v$. As a consequence, parsing reduces to a counting of positive and negative polarities present in the selected tagging for every pair (f, v): every positive occurrence counts for $+1$ and every negative occurrence for -1, the sum must be 0.
- Again in the tagging step, original methods were developped to filter out bad taggings. Each unsaturated polarity p in the grammar induces constraints on the set of contexts in which it can be used: the unsaturated polarity p must find a *companion* (*i.e.* a tree description able to saturated it); and the set of companions for the polarity p can be computed statically from the grammar. Each lexical selection which contains an unsaturated polarity without one of its companions can be safely removed.
- In the next step (node-merging phase), polarities are used to cut off parsing branches when their trees contain too many non neutral polarities.

5.1.2. Current state of the implementation

Leopar is presented and documented at http://leopar.loria.fr; an online demonstration page can be found at http://leopar.loria.fr/demo.

It is open-source (under the CECILL License http://www.cecill.info) and it is developed using the InriaGforge platform (http://gforge.inria.fr/projects/semagramme/)

The main features of current software are:

- automatic parsing of a sentence or a set of sentences,
- dependency and parse-tree representation of sentences,
- interactive parsing (the user chooses the couple of nodes to merge),
- visualization of grammars produced by XMG-2 or of sets of description trees associated to some word in the linguistic resources,

5.2. ACG Development Toolkit

Participants: Sylvain Pogodalla [correspondant], Philippe de Groote.

In order to support the theoretical work on ACG, we have been developing a support system. The objectives of such a system are twofold:

1. to make possible to implement and experiment grammars the modeling of linguistic phenomena;
2. to make possible to implement and experiment results related to the ACG formalisms. Such results can concern parsing algorithms, type extensions, language extensions, etc.

The ACG Development toolkit development effort is part of the POLYMNIE project (see Section 7.2.1.1 . It will support the experimentation and evaluation parts of the project.

The current version of the the ACG development toolkit prototype [1] issues from a first release published in October 2008. Further releases have been published before the ESSLLI 2009 course on ACG. It focuses on providing facilities to develop grammars. To this end, the type system currently implemented is the linear core system plus the (non-linear) intuitionistic implication, and a special attention has been paid to type error management. As a major limitation, this version only considers transformation from abstract terms to object terms, and not the other way around.

Enabling transformation from the object terms to the abstract terms is the first step of future development for the ACG support system. A parsing algorithm based on [32]'s methods is being implemented for second-order ACGs. It is based on a translation of ACG grammars into Datalog programs and is well-suited to fine-grained optimization.

However, since we're interested not only by recognizability (hence whether some fact is provable) but also by the parsing structure (hence the proof), the Datalog solver requires further adaptations. Note however that in the general case, the decidability of translating an object term to an abstract one is still an open problem.

5.3. Grew

Participants: Bruno Guillaume [correspondant], Guy Perrier.

Graph rewriting, Interface syntaxe-sémantique

Grew is a Graph Rewriting tools dedicated to applications in NLP. It is freely-available (from the page http://grew.loria.fr) and it is developed using the InriaGforge platform (http://gforge.inria.fr/projects/semagramme/)

We list below some of the major specificities of the GREW software.

- Graph structures can use a build-in notion of feature structures.
- The left-hand side of a rule is described by a graph called a pattern; injective graph morphisms are used in the pattern matching algorithm.
- Negative pattern can be used for a finer control on the left-hand side of rules.
- The right-hand side or rules is described by a sequence of atomic commands that describe how the graph should be modified during the rule application.
- Rules can be parametrized by lexical information.
- Filters can be used at the output of each module to control the structure produced are well-formed.
- Subset of rules are grouped in modules; the full rewriting process being a sequence of module applications.
- The Grew software has support both for confluent and non-confluent modules; when a non-confluent modules is used, all normal forms are returned and then ambiguity is handled in a natural way.
- Grew can be used on Corpus mode with statistics about rules usage or with an a Graphical User Interface which can show all intermediate graphs used during the rewriting process (useful either to debug rewriting system or for demonstrations).

[1] Available at http://acg.gforge.inria.fr with a CeCILL license.

A demonstration of the Grew Software was presented at the TALN conference in june in Grenoble.[15]

The Grew software were used for several kind of applications manipulating syntactic and/or semantic graph representations. It was used to build DMRS semantic representation from syntactic dependency trees in the French TreeBank [12], [14]. More recently, it was used on the Sequoia TreeBank, to produce deep syntax annotation and DMRS Semantic representations.

Another application of the Grew software which is currently investigated is the detection of annotation errors in corpora. Graph Rewriting is use to detect ill-formed structures that don't fit the annotation guide requirements. In collaboration with the Alpage team, this was applied to the Sequoia Corpus and the reported errors were corrected in version 3.2 and 3.3 of the corpus[2].

5.4. Other developments

Participant: Bruno Guillaume [correspondant].

Concordancer, Dependencies, Graphical tools Other peripheral developments of the team are available either as web service of as downloadable code:

- A concordancer named CONDOR which is usable online: http://condor.loria.fr. With Condor, ut is possible to search for all inflexions (given by a lexicon) of some lemma; it is possible to search for a couple of lemmas to find collocations.
- A program (named DEP2PICT) to build graphical representations (PNG, SVG or PDF) of dependency structures. It presented in http://dep2pict.loria.fr; it is usable online http://dep2pict.loria.fr/demo.

[2] https://www.rocq.inria.fr/alpage-wiki/tiki-index.php?page=CorpusSequoia