IN PARTNERSHIP WITH:
CNRS

Ecole normale supérieure de Paris

Activity Report 2016

Project-Team PARKAS

Parallélisme de Kahn Synchrone

IN COLLABORATION WITH: Département d'Informatique de l'Ecole Normale Supérieure

RESEARCH CENTER
Paris

THEME
Embedded and Real-time Systems

Table of contents

Project-Team PARKAS

Creation of the Team: 2011 April 01, updated into Project-Team: 2012 January 01

Keywords:

Computer Science and Digital Science:

1.1.1. - Multicore
1.1.3. - Memory models
2.1.1. - Semantics of programming languages
2.1.3. - Functional programming
2.1.6. - Concurrent programming
2.1.8. - Synchronous languages
2.2.2. - Memory models
2.2.3. - Run-time systems
2.2.4. - Parallel architectures
2.2.5. - GPGPU, FPGA, etc.
2.2.6. - Adaptive compilation
2.3. - Embedded and cyber-physical systems
2.3.1. - Embedded systems
2.3.2. - Cyber-physical systems
2.3.3. - Real-time systems
2.4.3. - Proofs
3.1.3. - Distributed data
3.1.8. - Big data (production, storage, transfer)
6.2.1. - Numerical analysis of PDE and ODE
6.2.7. - High performance computing

Other Research Topics and Application Domains:

5.2.1. - Road vehicles
5.2.2. - Railway
5.2.3. - Aviation
6.4. - Internet of things
6.6. - Embedded systems
9.2.1. - Music, sound
9.4.1. - Computer science
9.4.2. - Mathematics

1. Members

Research Scientists

Timothy Bourke [Inria, Starting Research position]
Albert Cohen [Inria, Senior Researcher, HDR]
Francesco Zappa Nardelli [Inria, Senior Researcher, HDR]

Faculty Member

Marc Pouzet [Team leader, Univ. Paris VI, Professor]

Engineers

Jean-Baptiste Brejon [Inria, until Sep 2016]
Michael Kruse [Inria]
Mircea Namolaru [Inria, until Sep 2016, granted by FP7 EuroLab-4-HPC project]
Laurent Morin [Inria, External collaborator from Rennes]
Zhen Zhang [Inria, Postdoctoral Fellow, until Jan 2016]
Oleksandr Zinenko [Inria, Postdoctoral Fellow]

PhD Students

Guillaume Baudart [ENS Paris]
Julien Proy [ENS Paris and INVIA, CIFRE Fellowship]
Adilla Susungi [Mines Paristech, PSL Fellowship]
Ulysse Beaugnon [ENS Paris]
Lelio Brun [ENS Paris, from Apr 2016]
Prasanth Chatarasi [Rice University and Inria, from May 2016 until Aug 2016]
Nhat Minh Le [ENS Paris and Inria, until Nov 2016, granted by Min. du Redressement Productif]
Robin Morisset [ENS Paris and Inria, granted by Google Inc]
Chandan Reddy Gopal [ENS Paris and Inria]
Jie Zhao [Chinese scholarship from NDSC]

Post-Doctoral Fellows

Adrien Guatto [ENS Paris and Inria, until Jan 2016]
Guillaume Iooss [ENS Paris]

Administrative Assistant

Anna Bednarik [Inria]

Others

Tommaso Borghesi [Inria, Master Student, until Jan 2016]
Keyur Joshi [IIT Hyderabad and Inria, Master Student, May 2016 until Jul 2016]
Lilia Otmane Cherif [UPMC and Inria, Master Student, from Apr 2016 until Aug 2016]
Sven Verdoolaege [Independent contractor, Polly Labs grant of ARM]

2. Overall Objectives

2.1. Overall Objectives

The research in PARKAS focuses on the design, semantics, and compilation of programming languages which allow going from parallel deterministic specifications to target embedded code executing on sequential or multi-core architectures. We are driven by the ideal of a mathematical and executable language used both to program and simulate a wide variety of systems, including real-time embedded controllers in interaction with a physical environment (e.g., fly-by-wire, engine control), computationally intensive applications (e.g., video), and compilers that produce provably correct and efficient code.

The team bases its research on the foundational work of Gilles Kahn on the semantics of deterministic parallelism, the theory and practice of synchronous languages and typed functional languages, synchronous circuits, modern (polyhedral) compilation, and formal models to prove the correctness of low level code running on weak-memory processors.

To realize our research program, we develop languages (LUCID SYNCHRONE, REACTIVEML, LUCY-N, ZELUS), compilers (PPCG), contributions to open-source projects (isl, LLVM, gcc), tools to study language semantics (Ott) and to test optimization compilers in the presence of threads (cmmtest), and formalizations in Interactive Theorem Provers of language semantics (Vélus, *n*-synchrony, quasi-synchrony). These software projects constitute essential "laboratories": they ground our scientific contributions, guide and validate our research through experimentation, and are an important vehicle for mutually beneficial and long standing collaborations with industry.

3. Research Program

3.1. Programming Languages for Cyber-Physical Systems

We study the definition of languages for reactive and Cyber-Physical Systems in which distributed control software interacts closely with physical devices. We focus on languages that mix discrete-time and continuous-time; in particular, the combination of synchronous programming constructs with differential equations, relaxed models of synchrony for distributed systems communicating via periodic sampling or through buffers, and the embedding of synchronous features in a general purpose ML language.

The synchronous language SCADE, [1] based on synchronous languages principles, is ideal for programming embedded software and is used routinely in the most critical applications. But embedded design also involves modeling the control software together with its environment made of physical devices that are traditionally defined by differential equations that evolve on a continuous-time basis and approximated with a numerical solver. Furthermore, compilation usually produces single-loop code, but implementations increasingly involve multiple and multi-core processors communicating via buffers and shared-memory.

The major player in embedded design for cyber-physical systems is undoubtedly SIMULINK, [2] with MODELICA[3] a new player. Models created in these tools are used not only for simulation, but also for test-case generation, formal verification, and translation to embedded code. That said, many foundational and practical aspects are not well-treated by existing theory (for instance, hybrid automata), and current tools. In particular, features that mix discrete and continuous time often suffer from inadequacies and bugs. This results in a broken development chain: for the most critical applications, the model of the controller must be reprogrammed into either sequential or synchronous code, and properties verified on the source model have to be reverified on the target code. There is also the question of how much confidence can be placed in the code used for simulation.

We attack these issues through the development of the ZELUS research prototype, industrial collaborations with the SCADE team at ANSYS/Esterel-Technologies, and collaboration with Modelica developers at Dassault-Systèmes and the Modelica association. Our approach is to develop a *conservative extension* of a synchronous language capable of expressing in a single source text a model of the control software and its physical environment, to simulate the whole using off-the-shelf numerical solvers, and to generate target embedded code. Our goal is to increase faithfulness and confidence in both what is actually executed on platforms and what is simulated. The goal of building a language on a strong mathematical basis for hybrid systems is shared with the Ptolemy project at UC Berkeley; our approach is distinguished by building our language on a synchronous semantics, reusing and extending classical synchronous compilation techniques.

Adding continuous time to a synchronous language gives a richer programming model where reactive controllers can be specified in idealized physical time. An example is the so called quasi-periodic architecture studied by Caspi, where independent processors execute periodically and communicate by sampling. We have applied ZELUS to model a class of quasi-periodic protocols and to analyze an abstraction proposed for model-checking such systems.

[1] http://www.esterel-technologies.com/products/scade-suite
[2] http://www.mathworks.com/products/simulink
[3] https://www.modelica.org

Communication-by-sampling is suitable for control applications where value timeliness is paramount and lost or duplicate values tolerable, but other applications—for instance, those involving video streams—seek a different trade-off through the use of bounded buffers between processes. We developed the *n*-synchronous model and the programming language LUCY-N to treat this issue.

3.2. Efficient Compilation for Parallel and Distributed Computing

We develop compilation techniques for sequential and multi-core processors, and efficient parallel run-time systems for computationally intensive real-time applications (e.g., video and streaming). We study the generation of parallel code from synchronous programs, compilation techniques based on the polyhedral model, and the exploitation of synchronous Single Static Assignment (SSA) representations in general purpose compilers.

We consider distribution and parallelism as two distinct concepts.

- Distribution refers to the construction of multiple programs which are dedicated to run on specific computing devices. When an application is designed for, or adapted to, an embedded multiprocessor, the distribution task grants fine grained—design- or compilation-time—control over the mapping and interaction between the multiple programs.
- Parallelism is about generating code capable of efficiently exploiting multiprocessors. Typically this amounts to maing (in)dependence properties, data transfers, atomicity and isolation explicit. Compiling parallelism translates these properties into low-level synchronization and communication primitives and/or onto a runtime system.

We also see a strong relation between the foundations of synchronous languages and the design of compiler intermediate representations for concurrent programs. These representations are essential to the construction of compilers enabling the optimization of parallel programs and the management of massively parallel resources. Polyhedral compilation is one of the most popular research avenues in this area. Indirectly, the design of intermediate representations also triggers exciting research on dedicated runtime systems supporting parallel constructs. We are particularly interested in the implementation of non-blocking dynamic schedulers interacting with decoupled, deterministic communication channels to hide communication latency and optimize local memory usage.

While distribution and parallelism issues arise in all areas of computing, our programming language perspective pushes us to consider four scenarios:

1. designing an embedded system, both hardware and software, and codesign;
2. programming existing embedded hardware with functional and behavioral constraints;
3. programming and compiling for a general-purpose or high-performance, best-effort system;
4. programming large scale distributed, I/O-dominated and data-centric systems.

We work on a multitude of research experiments, algorithms and prototypes related to one or more of these scenarios. Our main efforts focused on extending the code generation algorithms for synchronous languages and on the development of more scalable and widely applicable polyhedral compilation methods.

3.3. Validation and Proof of Compilers

Compilers are complex software and not immune from bugs. We work on validation and proof tools for compilers to relate the semantics of executed code and source programs. We develop techniques to formally prove the correctness of compilation passes for synchronous languages (Lustre), and to validate compilation optimization for C code in the presence of threads.

3.3.1. Lustre:

The formal validation of a compiler for a synchronous language (or more generally for a language based on synchronous block diagrams) promises to reduce the likelihood of compiler-introduced bugs, the cost of testing, and also to ensure that properties verified on the source model hold of the target code. Such a validation would be complementary to existing industrial qualifications which certify the development process and not the functional correctness of a compiler. The scientific interest is in developing models and techniques that both facilitate the verification and allow for convenient reasoning over the semantics of a language and the behavior of programs written in it.

3.3.2. C/C++:

The recently approved C11 and C++11 standards define a concurrency model for the C and C++ languages, which were originally designed without concurrency support. Their intent is to permit most compiler and hardware optimizations, while providing escape mechanisms for writing portable, high-performance, low-level code. Mainstream compilers are being modified to support the new standards. A subtle class of compiler bugs is the so-called concurrency compiler bugs, where compilers generate correct sequential code but break the concurrency memory model of the programming language. Such bugs are observable only when the miscompiled functions interact with concurrent contexts, making them particularly hard to detect. All previous techniques to test compiler correctness miss concurrency compiler bugs.

4. Highlights of the Year

4.1. Highlights of the Year

4.1.1. Awards

Marc Pouzet won the Inria/French Académie des Sciences/Dassault Systèmes Innovation award.

5. New Software and Platforms

5.1. Cmmtest

FUNCTIONAL DESCRIPTION

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

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

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

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

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

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

5.2. GCC

KEYWORDS: Compilation - Polyhedral compilation

FUNCTIONAL DESCRIPTION

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

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

5.3. Heptagon

FUNCTIONAL DESCRIPTION

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

- Participants: Adrien Guatto, Marc Pouzet, Cédric Pasteur, Léonard Gerard, Brice Gelineau, Gwenael Delaval and Eric Rutten
- Contact: Marc Pouzet

5.4. Lem

lightweight executable mathematics

FUNCTIONAL DESCRIPTION

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

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

5.5. Lucid Synchrone

FUNCTIONAL DESCRIPTION

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

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

5.6. Lucy-n

Lucy-n: an n-synchronous data-flow programming language
FUNCTIONAL DESCRIPTION

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

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

5.7. Ott

FUNCTIONAL DESCRIPTION

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

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

a Coq version of the definition,

an Isabelle version of the definition, and

a HOL version of the definition.

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

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

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

5.8. PPCG

FUNCTIONAL DESCRIPTION

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

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

5.9. ReactiveML

FUNCTIONAL DESCRIPTION

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

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

- Participants: Guillaume Baudart, Louis Mandel and Cédric Pasteur
- Contact: Guillaume Baudart
- URL: http://rml.lri.fr

5.10. SundialsML

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

Sundials/ML is an OCaml interface to the Sundials suite of numerical solvers (CVODE, CVODES, IDA, IDAS, KINSOL, ARKODE). It supports all features except for the Hypre and PETSC nvectors (which require additional libraries). Its structure mostly follows that of the Sundials library, both for ease of reading the existing documentation and for adapting existing source code, but several changes have been made for programming convenience and to increase safety, namely:

- solver sessions are mostly configured via algebraic data types rather than multiple function calls,
- errors are signalled by exceptions not return codes (also from user-supplied callback routines),
- user data is shared between callback routines via closures (partial applications of functions),
- vectors are checked for compatibility (using a combination of static and dynamic checks), and
- explicit free commands are not necessary since OCaml is a garbage-collected language.

FUNCTIONAL DESCRIPTION

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

This year we updated our interface to work with versions 2.6.0 and 2.7.0 of the Sundials library. This included significant work to support the new ARKODE solver, sparse matrices and the KLU and SuperLU/MT linear solvers, OpenMP and Pthreads nvectors, and various new functions and linear solvers in existing solvers. The source files were completely reorganized. The OCaml types for nvectors were adapted to support multiple nvectors. Memory leaks were eliminated and the performance problems investigated. This work was presented at the ACM Workshop on ML [28].

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

5.11. Zélus

SCIENTIFIC DESCRIPTION

The Zélus implementation has two main parts: a compiler that transforms Zélus programs into OCaml programs and a runtime library that orchestrates compiled programs and numeric solvers. The runtime can use the Sundials numeric solver, or custom implementations of well-known algorithms for numerically approximating continuous dynamics.
FUNCTIONAL DESCRIPTION

Zélus is a new programming language for hybrid system modeling. It is based on a synchronous language but extends it with Ordinary Differential Equations (ODEs) to model continuous-time behaviors. It allows for combining arbitrarily data-flow equations, hierarchical automata and ODEs. The language keeps all the fundamental features of synchronous languages: the compiler statically ensure the absence of deadlocks and critical races, it is able to generate statically scheduled code running in bounded time and space and a type-system is used to distinguish discrete and logical-time signals from continuous-time ones. The ability to combines those features with ODEs made the language usable both for programming discrete controllers and their physical environment.

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

5.12. isl

FUNCTIONAL DESCRIPTION

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

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

6. New Results

6.1. Verified compilation of Lustre

Participants: Timothy Bourke, Lélio Brun, Marc Pouzet.

Synchronous dataflow languages and their compilers are increasingly used to develop safety-critical applications, like fly-by-wire controllers in aircraft and monitoring software for power plants. A striking example is the SCADE Suite tool of ANSYS/Esterel Technologies which is DO-178B/C qualified for the aerospace and defense industries. This tool allows engineers to develop and validate systems at the level of abstract block diagrams that are automatically compiled into executable code.

Formal modelling and verification in an interactive theorem prover can potentially complement the industrial certification of such tools to give very precise definitions of language features and increased confidence in their correct compilation; ideally, right down to the binary code that actually executes.

This year we integrated elements of the CompCert verified C compiler into our Lustre compiler. In particular, we modularized the syntax and semantics of our source Lustre language and intermediate Obc language to be independent of the underlying types and operators of the host language. All previous proofs are independent of the choice of host language. We integrated CompCert by instantiating the types and operators with those of the Clight language and by adding a function that compiles an Obc program into Clight. The key challenge in this compilation pass is to move from a model where program variables are stored in a tree structure where distinctness is manifest to a model where variables are stored in nested structures in a single memory block with concomitant problems of aliasing, alignment, and memory size. We addressed this challenge by extending a CompCert library for expressing separation assertions and applying it to express our recursive predicates.

A similar approach was taken to address the encoding of multiple return values (permitted in Obc but not in Clight). We made various practical improvements to our compiler and proofs including the addition of a verified parser, the addition of an elaboration pass with type and clock checking, and pretty-printers for intermediate languages. It is now possible to compile scheduled and normalized Lustre programs to assembly code with a proof correction that relates the generated transition function to the dataflow semantics of the source program.

The initial part of this work, reported last year, has been published [20].

In collaboration with Pierre-Évariste Dagand (CNRS), Lionel Reig (Collège de France), and Xavier Leroy (Inria, GALLIUM team).

6.2. Fence Optimisations for Multicore Architectures

Participants: Robin Morisset, Francesco Zappa Nardelli.

We have pursued our investigation of sound optimisations for modern multicore architectures. Last year we focused on optimisations that can be expressed inside the semantics of the C11/C++11 programming language; we thus moved to optimisations that can be expressed only at the harware level. In particular we have shown how partial redundancy elimination (PRE) can be instantiated to perform *provably correct* fence elimination for multi-threaded programs running on top of the x86, ARM and IBM Power relaxed memory models. We have implemented our algorithm in the x86, ARM and Power backends of the `LLVM` compiler infrastructure. The optimisation does not induce an observable overhead at compile-time and can result in up-to 10% speedup on some benchmarks.

This work has been published in CC 2017 [18]. The implementation of the optimisations will be submitted for inclusion in the LLVM compiler suite.

6.3. Compiling synchronous languages for multi-processor implementations

Participants: Timothy Bourke, Albert Cohen, Guillaume Iooss, Marc Pouzet.

Working together with industrial partners in the context of the ASSUME project, we have been working to treat a large-scale and complete case study of an industrial application. This has involved studying the original sources and adapting the Heptagon Lustre compiler. Three main extensions have been developed this year: a mechanism to calculate and exploit module interdependencies; an extension to the type system to allow operator overloading via ad hoc polymorphism; and modifications to the parser to accept the provided source code. We have also worked on a means to generate dependency graphs from the provided nonfunctional specifications.

Our current work centers on understanding how to formalize the peculiarities of this class of application and the target architecture in our framework, and on generating Lustre code from the non-functional specifications. The ultimate aim is to generate correct multi-processor task-parallel real-time code for an embedded target and to integrate with both the Heptagon and Vélus compilers.

In collaboration (this year) with Dumitru Potop-Butucaru (Inria, AOSTE team), Keryan Didier (Inria, AOSTE team), Jean Souyris (Airbus), and Adrien Gauffriau (Airbus).

7. Bilateral Contracts and Grants with Industry

7.1. Bilateral Contracts with Industry

Technology Transfer Project, partly funded by the TETRACOM grant and by Kalray.

7.2. Bilateral Grants with Industry

Polly Labs initiative. Funded by ARM for 4 years with complementary support from Xilinx, in cooperation with ETH Zürich and Qualcomm.

8. Partnerships and Cooperations

8.1. National Initiatives

8.1.1. ANR

ANR WMC project (program “jeunes chercheuses, jeunes chercheurs”), 2012–2016, 200 Keuros. F. Zappa Nardelli is the main investigator.

ANR Boole project (program “action blanche”), 2009-2014.

ANR CAFEIN, 2013-2015. Marc Pouzet.

8.1.2. Investissements d’avenir

Sys2Soft contract (Briques Génériques du Logiciel Embarqué). Partenaire principal: Dassault-Systèmes, etc. Inria contacts are Benoit Caillaud (HYCOMES, Rennes) and Marc Pouzet (PARKAS, Paris).

ManycoreLabs contract (Briques Génériques du Logiciel Embarqué). Partenaire principal: Kalray. Inria contacts are Albert Cohen (PARKAS, Paris), Alain Darte (COMPSYS, Lyon), Fabrice Rastello (CORSE, Grenoble).

8.1.3. Others

Marc Pouzet is scientific advisor for the Esterel-Technologies/ANSYS company.

8.2. European Initiatives

8.2.1. FP7 & H2020 Projects

8.2.1.1. Eurolab-4-HPC

Title: EuroLab-4-HPC: Foundations of a European Research Center of Excellence in High Performance Computing Systems

Programm: H2020

Duration: September 2015 - September 2017

Coordinator: CHALMERS TEKNISKA HOEGSKOLA AB

Partners:

- Barcelona Supercomputing Center - Centro Nacional de Supercomputacion (Spain)
- Chalmers Tekniska Hoegskola (Sweden)
- Ecole Polytechnique Federale de Lausanne (Switzerland)
- Eidgenoessische Technische Hochschule Zuerich (Switzerland)
- Foundation for Research and Technology Hellas (Greece)
- Universitaet Stuttgart (Germany)
- Rheinisch-Westfaelische Technische Hochschule Aachen (Germany)
- Technion - Israel Institute of Technology (Israel)
- Universitaet Augsburg (Germany)
- The University of Edinburgh (United Kingdom)
- Universiteit Gent (Belgium)
- The University of Manchester (United Kingdom)

Inria contact: Albert Cohen

Europe has built momentum in becoming a leader in large parts of the HPC ecosystem. It has brought together technical and business stakeholders from application developers via system software to exascale systems. Despite such gains, excellence in high performance computing systems is often fragmented and opportunities for synergy missed. To compete internationally, Europe must bring together the best research groups to tackle the longterm challenges for HPC. These typically cut across layers, e.g., performance, energy efficiency and dependability, so excellence in research must target all the layers in the system stack. The EuroLab-4-HPC project's bold overall goal is to build connected and sustainable leadership in high-performance computing systems by bringing together the different and leading performance orientated communities in Europe, working across all layers of the system stack and, at the same time, fuelling new industries in HPC.

8.2.1.2. TETRACOM

Title: Technology Transfer in Computing Systems

Programm: FP7

Duration: September 2013 - August 2016

Coordinator: RHEINISCH-WESTFAELISCHE TECHNISCHE HOCHSCHULE AACHEN

Partners:

Imperial College of Science, Technology and Medicine (United Kingdom)

Rheinisch-Westfaelische Technische Hochschule Aachen (Germany)

Technische Universiteit Delft (Netherlands)

Tty-Saatio (Finland)

Universita di Pisa (Italy)

Inria contact: Albert Cohen

The mission of the TETRACOM Coordination Action is to boost European academia-to-industry technology transfer (TT) in all domains of Computing Systems. While many other European and national initiatives focus on training of entrepreneurs and support for start-up companies, the key differentiator of TETRACOM is a novel instrument called Technology Transfer Project (TTP). TTPs help to lower the barrier for researchers to make the first steps towards commercialisation of their research results. TTPs are designed to provide incentives for TT at small to medium scale via partial funding of dedicated, well-defined, and short term academia-industry collaborations that bring concrete R&D results into industrial use. This will be implemented via competitive Expressions-of-Interest (EoI) calls for TTPs, whose coordination, prioritization, evaluation, and management are the major actions of TETRACOM. It is expected to fund up to 50 TTPs. The TTP activities will be complemented by Technology Transfer Infrastructures (TTIs) that provide training, service, and dissemination actions. These are designed to encourage a larger fraction of the R&D community to engage in TTPs, possibly even for the first time. Altogether, TETRACOM is conceived as the major pilot project of its kind in the area of Computing Systems, acting as a TT catalyst for the mutual benefit of academia and industry. The projects primary success metrics are the number and value of coordinated TTPs as well as the amount of newly introduced European TT actors. It is expected to acquire around more than 20 new contractors over the project duration. TETRACOM complements and actually precedes the use of existing financial instruments such as venture capital or business angels based funding.

8.2.2. Collaborations in European Programs, Except FP7 & H2020

Program: ITEA 3

Project acronym: ASSUME

Project title: Affordable Safe & Secure Mobility Evolution

Duration: Sep 2015–Aug 2018

Coordinator: Udo Gleich

Other partners: AbsInt Angewandte Informatik GmbH, Airbus, Arcelik, Articus Systems AB, BTC Embedded Systems AG, Berner & Mattner Systemtechnik GmbH, Daimler AG, Eindhoven University of Technology, Ericsson, ANSYS, FindOut Technologies AB,

Ford Otosan, Forschungszentrum Informatik (FZI), Havelsan, KTH (Royal Institute of Technology), Kalray SA, Karlsruhe Institute of Technology (KIT), Kiel University, Koc University, KoçSistem, Model Engineering Solutions GmbH, Mälardalen University, NXP Semiconductors, OFFIS, Recore Systems BV, Robert Bosch GmbH, Safran Aircraft Engines SAS, Safran Electronics & Defense, Scania, TNO, Thales, UNIT Information Technologies R&D Ltd., University Pierre et Marie Curie, University of Technology in Munich, University of Twente, VDL Bus & Coach bv, Verum Software Tools BV, École normale supérieure.

Abstract: Future mobility solutions will increasingly rely on smart components that continuously monitor the environment and assume more and more responsibility for a convenient, safe and reliable operation. Currently the single most important roadblock for this market is the ability to come up with an affordable, safe multi-core development methodology that allows industry to deliver trustworthy new functions at competitive prices. ASSUME will provide a seamless engineering methodology, which addresses this roadblock on the constructive and analytic side.

8.2.3. Collaborations with Major European Organizations

Albert Cohen is an external member of the ARTEMIS-IA Working Group. Collaborating on the writing of the association’s Strategic Research Agenda (SRA), and the ECSEL JU Multi-Annual Research and Innovation Agenda (MASRIA).

https://artemis-ia.eu

8.3. International Initiatives

8.3.1. POLYFLOW

Title: Polyhedral Compilation for Data-Flow Programming Languages

International Partner (Institution - Laboratory - Researcher):

IISc Bangalore (India) - Department of Computer Science and Automation (CSA) - Uday Kumar Reddy Bondhugula

Start year: 2016

See also: http://polyflow.gforge.inria.fr

The objective of the associate team is to foster collaborations on fundamental and applied research. It also supports training sessions, exchange of undergraduate and master students, and highlighting opportunities in the partners’ research, education and economic environments.

Polyhedral techniques for program transformation are now used in several proprietary and open source compilers. However, most of the research on polyhedral compilation has focused on imperative languages, where computation is specified in terms of computational statements within nested loops and control structures. Graphical data-flow languages, where there is no notion of statements or a schedule specifying their relative execution order, have so far not been studied using a powerful transformation or optimization approach. These languages are extremely popular in the system analysis, modeling and design of embedded reactive control applications. They also underline the construction of domain-specific languages and compiler intermediate representations. The execution semantics of data-flow languages impose a different set of challenges for compilation and optimization. We are studying techniques enabling the extraction of a polyhedral representation from data-flow programs, to transform them with the goal of generating memory-efficient and high-performance code for modern architectures.

The research conducted in PolyFlow covers both fundamental and applied aspects. The partners also emphasize the development of solid research tools. The associate team will facilitate their dissemination as free software and their exploitation through industrial collaborations.

8.3.2. Inria International Partners

8.3.2.1. Informal International Partners

Pr. Peter Sewell, Computer Laboratory, University of Cambridge, UK. Regular visits and scientific collaboration.

Pr. Jan Vitek, College of Computer & Information Science Northeastern University, USA. Regular visits and scientific collaboration.

Prof. Uday Bondhugula, CSA department, Indian Institute of Science, India. See POLYFLOW associate team for details.

Prof. Ramakrishna Updadrasta, IIT Hyderabad, India, collaboration visits including internships.

Prof. P. Sadayappan, CS department, Ohio State University, USA. Joint publications, frequent visits, occasionally for several weeks.

Prof. M. Sheeran, Computer Science and Engineering Department, Chalmers University of Technology, Sweden. Regular visits. Continuing exchanges on languages and compilation for synchronous and hybrid systems.

Prof. C. Tinelli, CS department, University of IOWA, USA. Regular visits. Continuing exchanges on the verification of synchronous languages and programs.

Prof. R. von Hanxleden, Director at the Department of Computer Science, Head of the Real-Time and Embedded Systems Group, Kiel University, Germany. Regular visits and scientific collaboration.

Prof. M. Mendler, Head of the Informatics Theory Group, Bamberg University, Germany. Regular visits and scientific collaboration.

Dr. Sven Verdoolaege, CS department, K. U. Leuven, Belgium. Joint steering of the Polly Labs initiative and contractual cooperation in this context.

Dr. Tobias Grosser in the group of Prof. Torsten Hoeffler, ETH Zürich. Joint steering of the Polly Labs initiative. See Polly Labs for details.

8.4. International Research Visitors

8.4.1. Visits of International Scientists

8.4.1.1. Internships

Prasanth Chatarasi, PhD student from Rice University.

Keyur Joshi, undergraduate student from IIT Hyderabad.

8.4.2. Visits to International Teams

8.4.2.1. Research Stays Abroad

Guillaume Baudart spent three months working at the IBM Thomas J. Watson Research Centre.

9. Dissemination

9.1. Promoting Scientific Activities

9.1.1. Scientific Events Organisation

9.1.1.1. General Chair, Scientific Chair

- Albert Cohen is the General Chair of PLDI 2017

9.1.1.2. Member of the Conference Program Committees

- Timothy Bourke was a member of the PC of EMSOFT 2016.
- Francesco Zappa Nardelli was a member of the PC of POPL 2017.
- Francesco Zappa Nardelli will be a member of the PC of ECOOP 2017.
- Albert Cohen was a PC member of ASPLOS, PACT, PPoPP, CGO, PLDI.

9.1.1.3. Reviewer

- Timothy Bourke was a reviewer for FM 2016 (Int. Symposium on Formal Methods).

9.1.2. Journal

9.1.2.1. Member of the Editorial Boards

- Albert Cohen is an Associate Editor of ACM TACO.

9.1.2.2. Reviewer - Reviewing Activities

- Timothy Bourke was a reviewer for IEEE Embedded Systems Letters, ACM Transactions on Embedded Computing Systems, and IEEE Transactions on Software Engineering.

9.1.3. Invited Talks

- April, T. Bourke presented “Towards the verified compilation of Lustre” in the Gallium seminar series in Paris, France.
- December, T. Bourke presented “Verifying a Lustre Compiler (Part 1)” at the SYNCHRON workshop in Bamberg, Germany.
- November, F. Zappa Nardelli presented “Shared Memory Concurrency and Compiler Optimisations” at IMDEA, Madrid, Spain.

9.2. Teaching - Supervision - Juries

9.2.1. Teaching

Master: F. Zappa Nardelli: “A Programmer’s introduction to Computer Architectures and Operating Systems" (M1), 45h, École Polytechnique, France

Master: A. Cohen & F. Zappa Nardelli, “Semantics, languages and algorithms for multicore programming", Lecture, 9h+12h, M2, MPRI: Ecole normale supeérieure and Université Paris Diderot, France

Licence: F. Zappa Nardelli: “Conception et analyse d’algorithmes" (L3), PCs, 32h, E´cole Polytechnique, France

Master : M. Pouzet & T. Bourke: “Synchronous Systems” (M2), Lectures and TDs, MPRI, France

Master: T. Bourke participated in reviewing the M1 internships of students at the ENS, France.

Licence : M. Pouzet & T. Bourke: “Operating Systems” (L3), Lectures and TDs, ENS, France.

Licence : T. Bourke, “Digital Systems” (L3), Lectures and TDs, ENS, France

Marc Pouzet is Director of Studies for the CS department, at ENS.

9.2.2. Supervision

PhD in progress : Ulyssse Beaugnon, 2nd year, supervised by A. Cohen and M. Pouzet.

PhD in progress : Chandan Reddy, 2nd year, supervised by A. Cohen.

PhD in progress : Jie Zhao, 2nd year, supervised by A. Cohen.

PhD in progress : Guillaume Baudart, 3rd year, supervised by T. Bourke and M. Pouzet. This thesis will be defended in March.

PhD in progress : Lélio Brun, 1st year, supervised by T. Bourke and M. Pouzet.

PhD in progress : Robin Morisset, 3rd year, supervised by F. Zappa Nardelli. This thesis will be defended in April 2017.

9.2.3. Juries

Francesco Zappa Nardelli was an external reviewer of the PhD thesis of Carl Leonardsson, Uppsala University, Sweden.

Francesco Zappa Nardelli was a jury member of the PhD thesis of Nhat Minh Lê, ENS, Paris, France.

Timothy Bourke was an external reviewer of the masters thesis of Shruti Saini, The University of the South Pacific.

10. Bibliography

Publications of the year

Articles in International Peer-Reviewed Journals

[1] A. ACHARYA, U. BONDHUGULA, A. COHEN. *Polyhedral Auto-Transformation almost without ILP*, in "ACM Transactions on Programming Languages and Systems (TOPLAS)", May 2016, https://hal.inria.fr/hal-01425546

[2] G. BAUDART, A. BENVENISTE, T. BOURKE. *Loosely Time-Triggered Architectures*, in "ACM Transactions on Embedded Computing Systems (TECS)", August 2016, vol. 15, Article 71 [*DOI :* 10.1145/2932189], https://hal.inria.fr/hal-01408224

[3] S. G. BHASKARACHARYA, U. BONDHUGULA, A. COHEN. *Automatic Storage Optimization for Arrays*, in "ACM Transactions on Programming Languages and Systems (TOPLAS)", 2016, Original submission, candidate for presentation at PLDI 2016, https://hal.archives-ouvertes.fr/hal-01257223

[4] S. G. BHASKARACHARYA, U. BONDHUGULA, A. COHEN. *Automatic Storage Optimization for Arrays*, in "ACM Transactions on Programming Languages and Systems (TOPLAS)", 2016, vol. 38, pp. 1 - 23 [*DOI :* 10.1145/2845078], https://hal.inria.fr/hal-01425564

[5] U. BONDHUGULA, A. ACHARYA, A. COHEN. *The Pluto+ Algorithm: A Practical Approach for Parallelization and Locality Optimization of Affine Loop Nests*, in "ACM Transactions on Programming Languages and Systems (TOPLAS)", 2016, https://hal.archives-ouvertes.fr/hal-01257226

[6] T. BOURKE, R. J. VAN GLABBEEK, P. HÖFNER. *Mechanizing a Process Algebra for Network Protocols*, in "Journal of Automated Reasoning", March 2016, vol. 56, pp. 309-341 [*DOI :* 10.1007/s10817-015-9358-9], https://hal.inria.fr/hal-01408217

[7] I. LLOPARD, C. FABRE, A. COHEN. *A From a Formalized Parallel Action Language to its Efficient Code Generation*, in "ACM Transactions on Embedded Computing Systems (TECS)", January 2017 [*DOI :* 10.1145/0000000.0000000], https://hal.inria.fr/hal-01425140

International Conferences with Proceedings

[8] G. BAUDART, T. BOURKE, M. POUZET. *Soundness of the Quasi-Synchronous Abstraction*, in "Formal Methods in Computer-Aided Design (FMCAD)", Mountain View, CA, United States, Proceedings of the 16th International Conference on Formal Methods in Computer-Aided Design, October 2016, pp. 9-16, https://hal.inria.fr/hal-01408208

[9] S. G. BHASKARACHARYA, U. BONDHUGULA, A. COHEN. *SMO: An Integrated Approach To Intra-Array and Inter-Array Storage Optimization*, in "Symp. on Principles of Programming Languages (POPL)", St Petersburg, FL, United States, 2016, https://hal.archives-ouvertes.fr/hal-01257228

[10] S. G. BHASKARACHARYA, U. BONDHUGULA, A. COHEN. *SMO: An Integrated Approach to Intra-array and Inter-array Storage Optimization*, in "POPL 2016 - ACM Symposium on Principles of Programming Languages", Saint Petersburg, United States, January 2016, pp. 526-538 [*DOI :* 10.1145/2837614.2837636], https://hal.inria.fr/hal-01425888

[11] A. COHEN, A. DARTE, P. FEAUTRIER. *Static Analysis of OpenStream Programs*, in "6th International Workshop on Polyhedral Compilation Techniques (IMPACT'16), held with HIPEAC'16", Prague, Czech Republic, Proceedings of the IMPACT series, Michelle Strout and Tomofumi Yuki, January 2016, https://hal.inria.fr/hal-01251845

[12] A. COHEN, V. PERRELLE, D. POTOP-BUTUCARU, M. POUZET, E. SOUBIRAN, Z. ZHANG. *Hard Real Time and Mixed Time Criticality on Off-The-Shelf Embedded Multi-Cores*, in "International Conference on Embedded and Real-Time Software and Systems (ERTS2)", Toulouse, France, January 2016, https://hal.inria.fr/hal-01425887

[13] X. K. DO, S. LOUISE, A. COHEN. *Transaction Parameterized Dataflow: A Model for Context-Dependent Streaming Applications*, in "Design, Automation & Test in Europe Conference & Exhibition (DATE)", Dresden, Germany, March 2016, https://hal.inria.fr/hal-01425902

[14] A. DREBES, J.-B. BRÉJON, A. POP, K. HEYDEMANN, A. COHEN. *Language-Centric Performance Analysis of OpenMP Programs with Aftermath*, in "International Workshop on OpenMP (IWOMP)", Nara, Japan, October 2016, pp. 237 - 250 [*DOI :* 10.1007/978-3-319-45550-1_17], https://hal.inria.fr/hal-01425903

[15] A. DREBES, A. POP, K. HEYDEMANN, A. COHEN. *Interactive visualization of cross-layer performance anomalies in dynamic task-parallel applications and systems*, in "IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)", Uppsala, Sweden, April 2016, pp. 274 - 283 [*DOI :* 10.1109/ISPASS.2016.7482102], https://hal.inria.fr/hal-01425892

[16] A. DREBES, A. POP, K. HEYDEMANN, A. COHEN, N. DRACH. *Scalable Task Parallelism for NUMA: A Uniform Abstractionfor Coordinated Scheduling and Memory Management*, in "PACT : International Conference on Parallel Architectures and Compilation", Haifa, Israel, ACM New York, NY, USA, September 2016, pp. 125-137 [*DOI :* 10.1145/2967938.2967946], http://hal.upmc.fr/hal-01365718

[17] F. GINDRAUD, F. RASTELLO, A. COHEN, F. BROQUEDIS. *A bounded memory allocator for software-defined global address spaces*, in "ISMM 2016 - 2016 ACM SIGPLAN International Symposium on Memory Management", Santa Barbara, United States, June 2016, https://hal.inria.fr/hal-01412919

[18] R. MORISSET, F. ZAPPA NARDELLI. *Partially Redundant Fence Elimination for x86, ARM and Power processors*, in "International Conference on Compiler Construction (CC)", Austin, United States, February 2017, https://hal.inria.fr/hal-01423612

[19] S. VERDOOLAEGE, A. COHEN. *Live Range Reordering*, in "6þ Workshop on Polyhedral Compilation Techniques (IMPACT, associated with HiPEAC)", Prag, Czech Republic, 2016, https://hal.archives-ouvertes.fr/hal-01257224

National Conferences with Proceedings

[20] T. BOURKE, P.-E. DAGAND, M. POUZET, L. RIEG. *Vérification de la génération modulaire du code impératif pour Lustre*, in "JFLA 2017 - Vingt-huitième Journées Francophones des Langages Applicatifs", Gourettes, France, January 2017, https://hal.inria.fr/hal-01403830

Conferences without Proceedings

[21] A. COHEN, V. PERRELLE, D. POTOP-BUTUCARU, M. POUZET, E. SOUBIRAN, Z. ZHANG. *Hard Real Time and Mixed Time Criticality on Off-The-Shelf Embedded Multi-Cores*, in "8th European Congress on Embedded Real Time Software and Systems (ERTS 2016)", Toulouse, France, January 2016, https://hal.archives-ouvertes.fr/hal-01259157

[22] A. DREBES, J.-B. BRÉJON, A. POP, K. HEYDEMANN, A. COHEN. *Language-Centric Performance Analysis of OpenMP Programs with Aftermath*, in "International Workshop on OpenMP", Nara, Japan, October 2016, http://hal.upmc.fr/hal-01343686

[23] C. HONG, W. BAO, A. COHEN, S. KRISHNAMOORTHY, L.-N. POUCHET, F. RASTELLO, J. RAMANUJAM, S. PONNUSWANY. *Effective padding of multidimensional arrays to avoid cache conflict misses*, in "PLDI 2016: Proceedings of the 37th ACM SIGPLAN Conference on Programming Language Design and Implementation", Santa Barbara, United States, June 2016, https://hal.inria.fr/hal-01335346

Scientific Books (or Scientific Book chapters)

[24] S. POP, A. COHEN. *SSA-based Compiler Design*, in "SSA-based Compiler Design", F. RASTELLO (editor), springer, August 2016, vol. Loop tree and induction variables, ISBN 978-1-4419-6201-0, https://hal.archives-ouvertes.fr/hal-01257229

Research Reports

[25] D. BARTHOU, G. GROSDIDIER, K. PETROV, M. KRUSE, C. EISENBEIS, O. PÈNE, O. BRAND-FOISSAC, C. TADONKI, R. DOLBEAU. *Automated Code Generation for Lattice QCD Simulation*, University of Bordeaux, University of Paris Sud, Inria, University of Paris Sud, Mines ParisTech, CAPS Entreprise, June 2016, https://hal-mines-paristech.archives-ouvertes.fr/hal-01433302

[26] A. BENVENISTE, B. CAILLAUD, M. POUZET, H. ELMQVIST, M. OTTER. *Structural Analysis of Multi-Mode DAE Systems*, Inria, July 2016, n^{o} RR-8933, 32 p. , https://hal.inria.fr/hal-01343967

[27] A. COHEN, A. DARTE, P. FEAUTRIER. *Static Analysis of OpenStream Programs*, CNRS ; Inria ; ENS Lyon, January 2016, n^{o} RR-8764, 26 p. , Corresponding publication at IMPACT'16 (http://impact.gforge.inria.fr/impact2016), https://hal.inria.fr/hal-01184408

Other Publications

[28] T. BOURKE, J. INOUE, M. POUZET. *Sundials/ML: interfacing with numerical solvers*, September 2016, ACM Workshop on ML, https://hal.inria.fr/hal-01408230

[29] A. DREBES, A. POP, K. HEYDEMANN, N. DRACH, A. COHEN. *NUMA-aware scheduling and memory allocation for data-flow task-parallel applications*, ACM New York, NY, USA, March 2016, pp. 44:1-44:2, ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, Poster [*DOI :* 10.1145/2851141.2851193], http://hal.upmc.fr/hal-01365746