Proposal for a joint "équipe projet" between Inria and UCBL, integrated to UMR LIP

Analyses, Hardware/Software Compilation, Code Optimization for Complex Dataflow HPC Applications

Short temporary name: CASH (Compilation and Analyses for Software and Hardware)

Inria research field/theme: Algorithmics, Programming, Software and Architecture / Architecture, Languages and Compilation

Christophe Alias & Laure Gonnord & Matthieu Moy

November 22, 2017

1 Research Statement

The advent of parallelism in supercomputers and in more classical end-user computers increases the need for high-level code optimization and improved compilers.

Until 2006, the typical power-consumption of a chip remained constant for a given area as the transistor size decreased (this evolution is referred to as Dennard scaling). In other words, energy efficiency was following an exponential law similar to Moore's law. This is no longer true, hence radical changes are needed to further improve power efficiency, which is the limiting factor for large-scale computing. Improving the performance under a limited energy budget must be done by rethinking computing systems at all levels: hardware, software, compilers and runtimes.

One of the bottlenecks of performance and energy efficiency is data movements. The operational intensity must be optimized to avoid memory-bounded performance. Compiler analysis are strongly required to explore the trade-offs (operational intensity vs local memory size, operational intensity vs peak performance for reconfigurable circuits).

These issues are considered as one of the main challenges in the Hipeac roadmap [13] which, among others, cites the two major issues :

- Enhance the energy efficiency of the design of embedded systems, and especially the design of optimized specialized hardware.
- Invent techniques to "expose data movement in applications and optimize them at runtime and compile time and to investigate communication-optimized algorithms".

Parallelism based on dataflow is one way to tackle these two issues. A dataflow application is made of several actors that can perform computations and communicate with other actors. It can be implemented in several ways: as software running on a parallel general-purpose architecture or on accelerators like GPU or many-core, or as hardware implementation, possibly running on reconfigurable chips (FPGA).

The overall objective of the CASH team is to take advantage of the characteristics of the specific hardware (generic hardware, hardware accelerators or reconfigurable chips) to *compile energy efficient software and hardware*. The long-term objective is to provide solutions for the end-user developers to use at their best the huge opportunities of these emerging platforms.

LIP – UMR CNRS / ENS Lyon / UCB Lyon 1 / Inria – 69007 Lyon
E-mail : Christophe.Alias@inria.fr, Laure.Gonnord@univ-lyon1.fr, Matthieu.Moy@univ-lyon1.fr

In this project, we plan to work on:

- The design of dataflow-based intermediate representations, that are expressive enough and enable further optimizations (Section 1.1).
- The extensions of these intermediate representations to enable complex control flow and complex data structures, and the design of associated analysis for optimized code generation for multicore processors and accelerators (Section 1.2).
- The application of the two preceding activities on High Level Synthesis, with additional resource constraints (Section 1.3).
- A parallel and scalable simulation of Systems-on-Chips, which, combined with the preceding activity, will result in a complete workflow for circuit design (Section 1.4).

1.1 Dataflow models for HPC applications

The transverse theme of this proposal is the study of the dataflow model for parallel programs: the dataflow formalism expresses a computation on an infinite number of values, that can be viewed as successive values of a variable during time. A dataflow program is structured as a set of *communicating processes* that communicate values through *communicating buffers*.

Examples of dataflow languages include the synchronous languages Lustre and Signal, as well as SigmaC; the DPN representation [5] (data-aware process network) is an example of a dataflow intermediate representation for a parallelizing compiler.

The dataflow model, which expresses at the same time data parallelism and task parallelism, is in our opinion one of the best models for analysis, verification and synthesis of parallel systems. This model will be our favorite representation for our programs. Indeed, it shares the "equational" description of computation and data with the polyhedral model, and the static single assignment representation inside compilers. The dataflow formalism can be used both as a programming language and as an intermediate representation within compilers.

This topic is transverse to the proposal. While we will not a priori restrict ourselves to dataflow applications (we also consider approaches to optimize CUDA and OpenCL code for example), it will be a good starting point and a convergence point to all the members of the team.

Participants. Christophe Alias, Laure Gonnord, Matthieu Moy.

1.2 Compiler algorithms and tools for irregular applications

The design and implementation of efficient compilers becomes more difficult each day, as they need to bridge the gap between *complex languages* and *complex architectures*. On one hand, high-level programming languages tend to become more distant from the hardware which they are meant to command. Application developers use languages that bring them close to the problem that they need to solve.

This topic is closely linked to Section 1.1 since the design of an efficient intermediate representation is made while regarding the analyzes it enables. The intermediate representation should be expressive enough to embed maximal information; however if the representation is too complex the design of scalable analyzes would be harder.

The dataflow model alone is not capable of expressing fine-grain parallelism such as instruction-level parallelism. For this, we advocate for the use of another intermediate representation for the analyses of dataflow blocks code as well as communicating buffers.

We consider two categories of approaches: heavyweight analysis and optimizations like the polyhedral model that can be applied locally to small compute-intensive kernels, and low-cost analysis that scale to very large programs to allow global optimization and diagnostic.

The polyhedral model focuses on regular programs, whose execution trace are predictable statically. Unfortunately, most of the algorithms used in scientific computing do not fit totally in this category.

We plan to explore the extensions of polyhedral techniques to handle irregular programs with while loops and complex data structures (such as trees and lists).

To extend the applicability of polyhedral-like analysis, we plan to incorporate new ideas coming from the abstract interpretation community: control flow, approximations, and also shape analysis; and from the termination community: rewriting is one of the major techniques that are able to handle complex data structures and also recursive programs.

To develop low-cost and scalable analysis, we plan to cross fertilize ideas coming from the abstract interpretation community as well as language design, dataflow semantics, and WCET estimation techniques. We already have experience in designing low-cost semi relational abstract domains for pointers [19, 15], as well as tailoring static analyses for specialized applications in compilation [12, 25], Synchronous Dataflow scheduling [24] and extending the polyhedral model to irregular applications [2].

Participants. Christophe Alias, Laure Gonnord, Matthieu Moy.

Targeted applications.

- Dataflow programs (SigmaC [6]) for which the fine grain parallelism is not taken into account.
- Recursive programs operating on arrays, lists and trees.
- Worklist algorithms: lists are not handled within the polyhedral domain.
- Generalist programs with complex behaviors: detecting non licit memory accesses, memory consumption, hotspots, ...
- Functional properties for large programs.
- GPGPU programs where we want to optimize copies from the global memory to the block kernels, to perform less data accesses and change data layout to improve locality.
- Dataflow programs with arrays and iterators operating on arrays.

Expected impact. The impact of this work is the significantly widened applicability of various tools/compilers related to parallelization: allow optimizations for a larger class of programs, and allow low-cost analysis that scale to very large programs.

We target both analysis for optimization and analysis to detect, or prove the absence of bugs.

1.3 Compiler Algorithms, Simulation and Tools for Reconfigurable Circuits

Reconfigurable circuits (Field Programmable Gate Arrays, FPGA) are now a credible solution for low-energy HPC. An FPGA chip can deliver the same computing power as a GPU for 10× less energy budget. Major companies (including Intel, Google, Facebook and Microsoft) show a growing interest for FPGA and are pushing for programming languages and compilers (High-Level Synthesis, HLS) for FPGA.

Our final goal is to compile a dataflow representation optimized for multiple criteria (throughput, energy consumption, FPGA resources) (*front-end*). In turn, this dataflow representation will be mapped on the reconfigurable chip (*back-end*). CASH will only focus on front-end algorithms. Producing an end-to-end compiler chain for FPGA is out of our scope. Our front-end compiler algorithms will be implemented as source-to-source transformations for HLS compilers (e.g. Intel HLS compiler, Xilinx VivadoHLS) which will produce the final FPGA configuration. Several issues must be tackled before designing a compiler algorithm: which target architecture? Which cost model?

An architecture template will be designed to fit FPGA resources. Our architecture template will strongly depend on the optimizations applied by the front-end (parallelism, I/O, throughput). Both should be studied at the same time. Dataflow architectures are a natural candidate, but adjustments

are required to fit FPGA constraints (2D circuit, few memory blocks). Ideas from systolic arrays [23] can be borrowed, despite the limitation to regular kernels and the lack of I/O flexibility. A trade-off must be found between pure dataflow and systolic communications. Also, irregular applications may require efficient speculation mechanisms.

A cost model for our architecture template will be defined to guide our compiler optimizations. As mentioned earlier, external data transfers are the main limiting factor for performances. The roofline model [27] must be redefined in light of FPGA constraints. Indeed, the peak performance is no longer constant: it depends on the optimization itself. The more operational intensity we get, the more local memory we use, the less parallelization we get (since FPGA resources are limited) and finally the less performances we get. At the same time, operational intensity allows to get rid of data transfers and to get more performances. Hence, an optimum must be found.

Finally, compiler algorithms must be designed to produce the architecture template while optimizing the cost model. We plan to invent new algorithms and to extend/modify HPC compiler algorithms. For example, but not exclusively, those of the polyhedral model. Our program model will include: regular constructions (for loops, array, affine constraints) as well as irregular constructions found in real life HPC kernels (early exits, while loops, indirect array accesses). This is mandatory to handle iterative kernels and sparse matrices, which are extensively used in HPC. Handling irregular constructions fundamentally change the way we think compiler algorithms: which iteration domain? Which schedule? Which target architecture? This topic will cross fertilize with topic 1.2 (Compiler algorithms and tools for irregular applications).

We already have an experience in designing dataflow models, scheduling and resource allocation algorithms, and data transfer optimization for HLS [5, 4, 1, 3].

This activity will benefit from the "simulation" axis of the team.

Participants. Christophe Alias, Matthieu Moy.

Targeted applications. We will target HPC and big data kernels:

- HPC kernels: linear solvers, stencils, matrix factorizations, BLAS kernels, etc. Many kernels can be found in the Polybench/C benchmark suite [20]. The irregular versions can be found in [21].
- Big data kernels used in data center applications such as deep learning algorithms [14].

Expected Impact. The short term impact is to improve the power of HLS compilers by putting to work polyhedral optimizations through resource-aware / IO-aware automatic parallelization. As for the long-term impact, we believe that high-level synthesis can leverage the concepts involved in the irregular analysis described in section 1.2, thus extending profitably the scope of our compiler analysis. From an industrial point of view, we plan to transfer the results of this research to the XtremLogic start-up [28], co-founded by Christophe Alias and Alexandru Plesco.

1.4 Simulation of Systems on a Chip

One of the bottlenecks in complex Systems on a Chip (SoCs) design flow is the simulation speed: it is necessary to be able to simulate the behavior of a complete system, including software, before the actual chip is available. Raising the level of abstraction from Register Transfer Level to more abstract simulations like Transaction Level Modeling (TLM) [18] in SystemC [17] allowed gaining several orders of magnitude of speed. We are particularly interested in the loosely timed coding style where the timing of the platform is not modeled precisely, and which allows the fastest simulations. Still, SystemC implementations used in production are still sequential, and one more order of magnitude in simulation speed could be obtained with proper parallelization techniques.

Work on SystemC/TLM parallel execution is both an application of other work on parallelism in the team and a tool complementary to HLS presented in Section 1.3. Indeed, some of the parallelization techniques we develop in CASH could apply to SystemC/TLM programs. Conversely, a complete design-flow based on HLS often needs fast system-level simulation: the full-system usually contains both parts designed using HLS, handwritten hardware components and software.

Participants. Christophe Alias, Matthieu Moy.

Targeted applications.

- TLM models of SoCs including processors and hardware accelerators, written in a loosely timed coding style.
- Heterogeneous simulations including a SystemC/TLM part to model the numerical part of the chip, and other simulators to model physical parts of the system.

Expected Impact. The short term impact is the possibility to improve simulation speed with a reasonable additional programming effort. Automatic parallelization has been shown to be hard, if at all possible on loosely timed models [7]. We focus on semi-automatic approaches where the programmer only needs minor modifications to the programs to get significant speedup.

2 Research group

The members of the proposed team are (short bio available in appendix B):

- **Matthieu Moy**, HDR, "maître de conférences" UCBL since Sept 1st 2017 (`https://matthieu-moy.fr`). Proposed team leader.
- **Christophe Alias**, "chargé de recherche" Inria, (`http://perso.ens-lyon.fr/christophe.alias`)
- **Laure Gonnord**, HDR, "maître de conférences" UCBL (`http://laure.gonnord.org/pro/`).

3 Relationship with the XtremLogic startup

Christophe Alias has a strong relationship with XtremLogic, startup that he cofounded with one of his former Phd, Alexandru Plesco. XtremLogic develops a complete compiler from C to Verilog, which was initially developed in the context of the Phd of Alexandru Plesco. Christophe is the sole author of the polyhedral compiler that was partially transferred in XtremLogic in April 2014 under an "Inria exclusive License". Since this date, Christophe is linked to XtremLogic by a "convention de concours scientifique Inria", and spends 20% of his time at XtremLogic as "conseiller scientifique" (scientific consultant). He owns shares of the company.

Since 2014, two versions of the compiler coexist: the *startup* version, whose development is pursued by Alexandru Plesco and an engineer, and a *research* version, where Christophe develops his algorithms and tools for his own ongoing academic research. There is a clear distinction between these two software. If a new algorithm developed by Christophe in his research time would interest the startup, then the license contract would be amended.

Christophe Alias' project is to remain close to the XtremLogic startup and benefit from this proximity in his research, but Christophe has no plan to leave Inria. He sees XtremLogic as a way to drive his research, not a way to leave research for industry.

The CASH team as a whole will strongly benefit from this proximity, but the collaboration will be done respecting our "research" identity and taking into account potential conflicts of interests. We make a clear distinction between the work done by XtremLogic in an industrial context and the research done

in CASH.

The proximity with XtremLogic gives the CASH team indirect access to end-users, and allows the team to get a better understanding of the current needs and practical challenges in the area of High-Level Synthesis for High-Performance Computing. This will greatly help the CASH team in its "case study driven" approach. From a scientific point of view, even though XtremLogic is a commercial entity, there are still major scientific challenges to solve. The discussions between Christophe Alias and XtremLogic have already been fruitful and led to several publications.

From a practical point of view, the distinction between CASH and XtremLogic's research subjects and tools are:

- XtremLogic produces an industrial hardware synthesis tool that produces directly a circuit description at the RTL (Register Transfer Level) level (Verilog). Generating RTL code raises a lot of low-level issues that are out of the scope of the CASH team. CASH will only work on the first steps of HLS. CASH tools will be source-to-source compilers that will produce parallel C code that is meant to be used as input to an existing synthesis tools like VivadoHLS.

- XtremLogic targets regular code (i.e. `for` loops with affine bounds, arrays...), where the polyhedral model applies directly. Part of the research of CASH is to go beyond the polyhedral model and to deal with irregular applications (`while` loops with complex conditions, dynamic data-structures...).

- Obviously, XtremLogic targets and industrial quality compiler while the CASH team will only produce research prototypes.

We do hope that the research carried out in CASH will be of interest to XtremLogic, but an effective "contractualized" collaboration is constrained by legal and administrative aspects. For example, Christophe cannot legally write code on the industrial compiler, nor supervise students working for XtremLogic. It is possible at least in theory to continue the intellectual property transfer, by going through an amendment of the contract signed with Inria. In practice, we foresee most of the collaboration through informal discussions and common publications, that are not subject to the contractualized aspects.

These limitations in the collaboration and the fact that we work on different tools limits potential risks: the research activity of CASH will not *depend* on XtremLogic, although it will benefit from it.

The research results obtained by CASH will be published and can be used by everyone: there is no exclusive transfer other than the ones negotiated by contract with Inria.

4 Positioning and added value

The current project proposal intends to make progress in the quest for performance of dataflow programs in particular on FPGAs, building upon the expertise of its members on algorithm design and formal methods.

The research teams that work in these domains in the Inria ecosystem are thus coming from two "themes" inside "Algorithms, programs, software and architecture":

- Architecture, languages, compilation: especially CAMUS and CORSE.
- Embedded systems and real time: especially SPADES and PARKAS.

Some other Inria teams belonging to other themes have also some shared research themes: ANTIQUE, CELTIQUE, and SOCRATE. There are also national non-Inria teams whose topics are close to ours.

We believe that High-Level Synthesis (HLS) is an understudied topic. It is studied in CAIRN and TIMA-SLS, but both use different approach. One contribution of CASH is the cross-fertilization with the compilation and abstract interpretation communities.

Abstract interpretation is studied in many teams, including ANTIQUE and Verimag-PACSS. What distinguishes CASH from these teams is the focus on low-cost analysis to allow more applications to compilers.

The added value of CASH in the compilation domain is the static-analysis based approach. Unlike CORSE and CAMUS, we do not consider dynamic approach. We distinguish the analysis phase and the optimization phase and focus on the analysis more than the optimization.

CASH is complementary with many other teams with which we plan to collaborate. Inside LIP, we will collaborate with Avalon on HPC applications, with ROMA on scheduling, with PLUME on program semantics and with AriC on arithmetic operators. Outside LIP, SOCRATE could both provide us tools and expertise in arithmetic operators and use our tools to implement radio applications. We already collaborate with Verimag-PACS on abstract interpretation.

We already have industrial partnerships with STMicroelectronics on hardware simulation, with Kalray on dataflow programming for many-core and with XtremLogic on HLS, and obviously plan to continue these collaborations.

More details can be found in section A.

References

[1] Christophe Alias, Fabrice Baray, and Alain Darte. Bee+Cl@k: An implementation of lattice-based array contraction in the source-to-source translator Rose. In *ACM Conf. on Languages, Compilers, and Tools for Embedded Systems (LCTES'07)*, 2007.

[2] Christophe Alias, Alain Darte, Paul Feautrier, and Laure Gonnord. Multi-dimensional rankings, program termination, and complexity bounds of flowchart programs. In *International Static Analysis Symposium (SAS'10)*, 2010.

[3] Christophe Alias, Alain Darte, and Alexandru Plesco. Optimizing remote accesses for offloaded kernels: Application to high-level synthesis for FPGA. In *ACM SIGDA Intl. Conference on Design, Automation and Test in Europe (DATE'13)*, Grenoble, France, 2013.

[4] Christophe Alias, Bogdan Pasca, and Alexandru Plesco. FPGA-specific synthesis of loop-nests with pipeline computational cores. *Microprocessors and Microsystems*, 36(8):606–619, November 2012.

[5] Christophe Alias and Alexandru Plesco. Data-aware Process Networks. Research Report RR-8735, Inria - Research Centre Grenoble – Rhône-Alpes, June 2015.

[6] Pascal Aubry, Pierre-Edouard Beaucamps, Frédéric Blanc, Bruno Bodin, Sergiu Carpov, Loïc Cudennec, Vincent David, Philippe Doré, Paul Dubrulle, Benoît Dupont De Dinechin, François Galea, Thierry Goubier, Michel Harrand, Samuel Jones, Jean-Denis Lesage, Stéphane Louise, Nicolas Morey Chaisemartin, Thanh Hai Nguyen, Xavier Raynaud, and Renaud Sirdey. Extended Cyclostatic Dataflow Program Compilation and Execution for an Integrated Manycore Processor. In *Alchemy 2013 - Architecture, Languages, Compilation and Hardware support for Emerging ManYcore systems*, volume 18 of *Proceedings of the International Conference on Computational Science, ICCS 2013*, pages 1624–1633, Barcelona, Spain, June 2013.

[7] Denis Becker, Matthieu Moy, and Jérôme Cornet. Parallel Simulation of Loosely Timed SystemC/TLM Programs: Challenges Raised by an Industrial Case Study. *MDPI Electronics*, 5(2):22, 2016.

[8] Louis Besème. Vers une accélération matérielle de gnuradio. Master's thesis, INSA de Lyon, Département informatique, mars 2015. Encadrants : Tanguy Risset et Florent De Dinechin.

[9] Bruno Blanchet, Patrick Cousot, Radhia Cousot, Jérôme Feret, Laurent Mauborgne, Antoine Miné, David Monniaux, and Xavier Rival. A static analyzer for large safety-critical software. In *Programming Language Design and Implementation (PLDI)*, pages 196–207, 2003.

[10] Alban Bourge, Olivier Muller, and Frédéric Rousseau. Automatic high-level hardware checkpoint selection for reconfigurable systems. In *Field-Programmable Custom Computing Machines (FCCM), 2015 IEEE 23rd Annual International Symposium on*, pages 155–158. IEEE, 2015.

[11] Mickaël Dardaillon, Kevin Marquet, Tanguy Risset, Jérôme Martin, and Henri-Pierre Charles. A new compilation flow for software-defined radio applications on heterogeneous mpsocs. *TACO*, 13(2):19:1–19:25, 2016.

[12] Paul Feautrier, Abdoulaye Gamatié, and Laure Gonnord. Enhancing the Compilation of Synchronous Dataflow Programs with a Combined Numerical-Boolean Abstraction. *CSI Journal of Computing*, 1(4):8:86–8:99, 2012. RR version = http://hal.inria.fr/hal-00780521/en.

[13] Hipeac roadmap for high performance computing, `http://www.hipeac.net/system/files/hipeac_roadmap1_0.pdf`, 2013.

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[15] Maroua Maalej, Vitor Paisante, Pedro Ramos, Laure Gonnord, and Fernando Pereira. Pointer Disambiguation via Strict Inequalities. In *Code Generation and Optimisation*, Austin, United States, February 2017.

[16] David Monniaux and Laure Gonnord. Cell morphing: from array programs to array-free Horn clauses. In Xavier Rival, editor, *23rd Static Analysis Symposium (SAS 2016)*, Static Analysis Symposium, Edimbourg, United Kingdom, September 2016.

[17] Open SystemC Initiative. *IEEE 1666 Standard: SystemC Language Reference Manual*, 2011.

[18] Open SystemC Initiative (OSCI). *OSCI TLM-2.0 Language Reference Manual*, June 2008.

[19] Vitor Paisante, Maroua Maalej, Leonardo Barbosa, Laure Gonnord, and Fernando Magno Quintao Pereira. Symbolic Range Analysis of Pointers. In *International Symposium of Code Generation and Optmization*, pages 791–809, Barcelon, Spain, March 2016.

[20] Louis-Noël Pouchet. Polybench: The polyhedral benchmark suite. *URL: http://www. cs. ucla. edu/~ pouchet/software/polybench/[cited July,]*, 2012.

[21] William H Press, Saul A Teukolsky, William T Vetterling, and Brian P Flannery. Numerical recipes in c++. *The art of scientific computing*, 2015.

[22] Adrien Prost-Boucle, Olivier Muller, and Frédéric Rousseau. Fast and standalone design space exploration for high-level synthesis under resource constraints. *Journal of Systems Architecture*, 60(1):79–93, 2014.

[23] Patrice Quinton. Automatic synthesis of systolic arrays from uniform recurrent equations. *ACM SIGARCH Computer Architecture News*, 12(3):208–214, 1984.

[24] Hamza Rihani, Matthieu Moy, Claire Maiza, Robert I. Davis, and Sebastian Altmeyer. Response time analysis of synchronous data flow programs on a many-core processor. In *Proceedings of the 24th International Conference on Real-Time Networks and Systems*, RTNS '16, pages 67–76, New York, NY, USA, 2016. ACM.

[25] Henrique Nazaré Willer Santos, Izabella Maffra, Leonardo Oliveira, Fernando Pereira, and Laure Gonnord. Validation of Memory Accesses Through Symbolic Analyses. In *Proceedings of the 2014 ACM International Conference on Object Oriented Programming Systems Languages And Applications (OOPSLA'14)*, Portland, Oregon, United States, October 2014.

[26] Manuel Selva, Lionel Morel, and Kevin Marquet. numap: A portable library for low-level memory profiling. In *SAMOS*, pages 55–62. IEEE, 2016.

[27] Samuel Williams, Andrew Waterman, and David Patterson. Roofline: an insightful visual performance model for multicore architectures. *Communications of the ACM*, 52(4):65–76, 2009.

[28] Xtremlogic sas, `http://www.xtremlogic.com`, 2017.

Appendices

A Details on Positioning and Added Value

The current project proposal intends to make progress in the quest for performance of dataflow programs on FPGAs, building upon the expertise of its members on algorithm design and formal methods.

The research teams that work in these domains in the Inria ecosystem are thus coming from two "themes" inside "Algorithms, programs, software and architecture":

- Architecture, languages, compilation: especially CAMUS, CORSE, CAIRN and PACAP.
- Embedded systems and real time: especially SPADES and PARKAS.

Some other Inria teams belonging to other themes have also some shared research themes: ANTIQUE, CELTIQUE, and SOCRATE. There are also national non-Inria teams whose topics are close to ours.

The following section does not have the vocation to be exhaustive. We quickly summarize the differences and convergences with the closest teams who share common topics with us, with a particular focus on local (Lyon and Grenoble) teams.

A.1 Inside the LIP

LIP is composed of the following teams:

- AriC (Arithmetic and Computing)
- Avalon (Algorithms and Software Architectures for Distributed and HPC Platforms)
- DANTE (Dynamic Network)
- MC2 (Models of computation, Complexity, Combinatorics)
- PLUME (programs and proofs)

- ROMA (Resource Optimization: Models, Algorithms and Scheduling)

The main topics of CASH (compilation, abstract interpretation, high-level synthesis) are not in the scope of any other teams.

Teams with the strongest relationships with us are:

- AVALON: The AVALON team also targets HPC applications, with a focus on the design on programming models supporting many kinds of architectures. "The team focuses in particular in energy and data intensive application profiling and modelization, data management, component based application description, and application mapping and scheduling". AVALON's activities are complementary to us since we specifically target HPC kernels (or programs designed around HPC kernels) for which we want to aggressively compute/compile an optimized (software of hardware) version. Clearly, all applications cannot be statically scheduled (or at least, only part of some large applications can be statically scheduled), and we would need efficient algorithms for our optimized kernels to be scheduled with other (possibly unknown) tasks. CASH does not target dynamic approaches like profiling that are in the scope of AVALON. AVALON considers the assembly of software components at the application level, while CASH considers computation kernels, or assembly of a few computation kernels. In other words, what CASH considers as a program is what AVALON considers as a component.
- PLUME: The activities of PLUME on program proofs focus on semantics and typing. For example, abstract interpretation is not in the scope of PLUME. PLUME works on abstract formalisms (λ-calculus, π-calculus, automata) rather than targeting concrete programming languages. We will benefit of the Plume team proximity for our activities in expressing parallel languages semantics (our vision is however more focused on efficiency than expressivity). In the converse, PLUME's activity on program verification of parallel models could benefit from the formalization of new research problems coming from the CASH activities on optimizations.
- ROMA: Roma works on models, algorithms, and scheduling strategies at OS level. Roma's programming model considers tasks as black boxes (unlike CASH). The compilation or analyses of individual boxes are not under the scope of ROMA. However, we strongly believe that working on the scheduling of dataflow applications could lead to a better understanding of their characteristics that could benefit to ROMA in the form of more accurate task models. Also, on-the-fly compilation of tasks under resource constraint can enable new runtime scheduling strategies. These are more long term possible collaborations.

A.2 CORSE

A.2.1 Summary of CORSE's activities

CORSE's activity report is available here:

`http://raweb.inria.fr/rapportsactivite/RA2016/corse/corse.pdf`

The overall objective of CORSE[1] is to address the ever increasing complexity of applications as well as the complexity of emerging platforms by combining static and dynamic compilation techniques. The team focuses on the interaction of the program and the runtime. Its main activities are:

- The design of efficient runtimes via strong interactions with the compiler/debugger.
- Runtime verification through the generation of observers or enforcers for temporal-logic properties.

[1] All the text concerning CORSE has been proof-read, discussed and validated by F. Rastello, head of the team.

- The design of efficient compiler optimizations via the combination of static and dynamic analyses.

Only the last research direction is relevant for CASH, as we do not have any expert in debugging nor runtime. The activities of CORSE in this direction are:

- Compiler Architecture Design. The activity focus on the design of an efficient single intermediate representation that is capable of expressing the program semantics at multiple levels (control-flow, data dependencies, profiling information). The particular focus of this activity is the design of representations that are easily maintainable and refinable at each step of static and dynamic compilation, thus can handle "information telescoping". Participants: Fabrice Rastello, Florent Bouchez Tichadou.
- Combining Static with Dynamic Analysis, Static Compiler Optimization with Run-Time Decisions. The hope of this activity is to provide to compiler back-end the information required to perform optimizing transformations (such as combined scheduling, vectorization, register tiling, and register allocation/promotion). For that purpose, the team develops combinations of static analyses and code duplication (hybrid analyses), use intensive profiling information to gather pertinent information such as runtime dependencies, and use it to perform more accurate code optimization. Participants: Fabrice Rastello, Florent Bouchez Tichadou, Frédéric Desprez.

A.2.2 Positioning of CASH

CORSE's project intersects ours mostly with Section 1.2 for which we plan future collaborations described in Section A.2.3. There is a huge amount of work to understand irregular applications, find ways to reason on them and perform clever static and runtime optimizations, for which a single team will clearly not be sufficient.

CORSE's research on static compilation has two objectives. The first one is to estimate performance characteristics of the code (bandwidth and operational intensity estimation) so that to construct a performance model of it. The second is to be combined with dynamic analyses. While we share the idea that purely static approaches will not be the definitive solutions for HPC compilation, our approach will mainly focus on the characterization of technical and scientific limits of static compilation, with a "high level language" approach.

CORSE's activity on runtime property verification is completely different from the property verification targeted by CASH. CORSE focuses on generation of observers or enforcers from possibly complex temporal logic properties, and consider programs under verification as black-boxes. As opposed to that, we analyze programs and do not consider temporal-logic properties, but only invariants.

There is a simulation activity in CORSE as well as in the current CASH proposal, but the targets are not the same: in CASH we target SoCs and FPGAs, in CORSE the objective is to generate simulators from existing architectures' ISA (instruction set).

The code generation for FPGAs is not studied in CORSE. The dynamic analyses of CORSE target classic platforms (multiprocessors, accelerators GPU/XeonPhi). CORSE does not work on abstract interpretation.

A.2.3 Possible collaborations

The activities of the "hybrid" axis of CORSE are complementary to the ones CASH plans to work on. We plan to collaborate on the following topics:

- Static Analyses for dynamic compilation: CORSE's work on profiling is complementary to the fully static approach of CASH. Laure Gonnord and Fabrice Rastello already share a research project on

this topic within the PROSPIEL Inria associate team [2]: the idea is to use static analyses to reduce the number of parameters to profile while optimizing a given application.

- Dependence hybrid analysis for efficient scheduling: CASH shares with CORSE the idea that the polyhedral model is not sufficient to express all data dependencies in HPC programs. Our theoretical work on dependencies will most probably have an echo in the hybrid compilation part of the CORSE's project. Indeed, we will have to deal with over-approximations of these dependencies, and computing preconditions for a code to be optimized efficiently would clearly be a way to limit the impact of these over-approximations. In the other direction, code profiling and monitoring can give hints to static code optimizers. Fabrice Rastello is a collaborator in an ANR JCJC proposal led by Laure Gonnord in Spring 2017[3], where these topics are developed in the context of data structure optimization.
- CORSE's activities on performance evaluation and debugging can be applied to SystemC simulations. Actually, a european project (SIM-RIDER, on Parallel and Distributed Simulation Engines) has been submitted and passed the first selection, including Matthieu Moy and the CORSE team (J.-F. Méhaut).

A.3 SOCRATE

A.3.1 Summary of SOCRATE's activities

SOCRATE's activity report is available here:

`http://raweb.inria.fr/rapportsactivite/RA2016/socrate/socrate.pdf.`

The SOCRATE team is composed of 3 research axes: "Flexible Radio Front-End", "Multi-User Communications" and "Software Radio Programming Model" (also referred to as the "embedded" axis). Only the last one is relevant in a comparison with CASH (others are strongly oriented towards signal processing and information theory).

The activities of the "embedded" axis of SOCRATE are: [4]

- Low-power systems: impact of the capabilities of modern hardware on low-level software. A topic of growing interest is non-volatile RAM (NVRAM), which do not loose their content when powered off. This allows in particular the design of transiently powered systems able to snapshot relevant parts of their state before being powered off. NVRAM raises new challenges in memory management and in the design of operating systems. This topic is of growing importance in SOCRATE. It is supported among others by IPL ZEP[5] and an ARC6 regional Ph.D grant. Participants are: Kevin Marquet, Tanguy Risset, Guillaume Salagnac.
- Data-flow programming model: SOCRATE works on code generation flows using data-flow languages as input and on the monitoring of these programs. Work on code generation consider dataflow programs as a set of tasks that are mapped and scheduled on a parallel and possibly heterogeneous architecture (e.g. M. Dardaillon's Ph.D [11] targeting the Magali many-core processor).

[2] The two other senior participants of this project are Sylvain Collange, from the PACAP Inria team (Rennes) and Fernando Pereira, from the Compilation Lab of the University of Mineas Gerais, Brasil. The objective of this project is to develop compilation techniques that let developers code high-performance programs in high-level parallel languages such as OpenCL or NVIDIA C for CUDA, while taking maximum benefit from modern hardware.

[3] Other collaborators are Tomofumi Yuki, Inria Rennes; Carsten Fuhs, Univ Birbeck, UK; Lionel Morel, Insa Lyon, and Christophe Alias of CASH.

[4] All the text concerning SOCRATE has been proof-read, discussed and validated by T. Risset, head of the team.

[5] Inria Project Lab "Zero-power computation", `https://project.inria.fr/iplzep/teams/`

A large part of SOCRATE's work on data-flow consider dynamic aspects like performance monitoring, for example the numap [26] library used to identify the performance bottlenecks due to memory accesses at runtime. This activity's importance is decreasing in SOCRATE because Tanguy Risset and Kevin Marquet are strongly involved in the "low-power systems" activity, Lionel Morel is currently looking for a mobility, and the collaboration with CEA on Magali is over. SOCRATE is still interested in data-flow programming models, but currently has other priorities and would welcome another team working on the subject in Lyon. More details in section 6.3.1 of the SOCRATE activity report. Participants: Kevin Marquet, Lionel Morel, Tanguy Risset.

- Tools for FPGA development. SOCRATE's activities on FPGA development are centered around the design of efficient and precise arithmetic operators. A great success of SOCRATE is the tool FloPoCo, `http://flopoco.gforge.inria.fr/`, which allows generating arithmetic operators from a set of parameters (like precision in number of bits). SOCRATE also works on the design of numerical filters outside FloPoCo, and the integration of computing cores generated by FloPoCo in a High-Level Synthesis (HLS) flow. More details in sections 6.3.2 to 6.3.4 of the activity report. Participant: Florent de Dinechin.

A.3.2 Positioning of CASH

We share with the SOCRATE team the diagnostic that dataflow applications are an essential application domain for HPC techniques, and the fact that memory transfer issues are crucial.

However, we study dataflow applications with completely different approaches and different applications in mind. SOCRATE's work on dataflow applications focuses on dynamic monitoring and task mapping on possibly heterogeneous architectures. SOCRATE has no activity in program analysis, neither for formal verification nor for optimizing compilers. There is no current work on languages or intermediate representations in SOCRATE.

Another point where SOCRATE and CASH can be compared is hardware design. SOCRATE works on the design of efficient and precise arithmetic operators, targeting FPGA. This work targets specific operators and unlike the HLS approach followed in CASH, do not attempt to synthesize hardware circuits for general programs. Work was started to use the data-flow language GNU Radio to assemble hardware components (Internship report of Louis Besème in 2015 [8]), but in this case GNU Radio is used as an architecture description language and not as a fully-fledged programming language as it is the case with HLS approaches.

Obviously, SOCRATE and CASH target different application domains. The main target of the "embedded" axis of SOCRATE is embedded system, with an emphasis on low or ultra-low power systems. CASH targets primarily HPC applications. This has a strong impact on the scientific approach: for example memory-management techniques for transiently-powered systems are out of scope for CASH. The focus of optimization is also different: for example large matrix manipulation is a common operation in HPC, which motivates the importance of the polyhedral optimizations studied in CASH.

A.3.3 Possible collaborations

The activities of the "embedded" axis of SOCRATE are complementary to the one CASH plans to work on. The geographical proximity will allow collaboration on several topics such as:

- Data-flow programming models: SOCRATE's work on dynamic profiling of data-flow application is complementary to the static approach of CASH. Laure Gonnord already has close interactions with Lionel Morel on this topic (in 2017 they co-advise two students) and we plan to continue the

collaboration which, for the moment, mainly focuses on the extension of expressivity of dataflow languages (SigmaC, Lustre) to target more code optimizations.

- Efficient hardware generation flow: CASH's work on HLS uses imperative code as input, and takes care of splitting the code into a network of communicating processes. It needs hardware implementations of arithmetic operators, and could use the work of Florent Dinechin for that. In other words, CASH works on the compiler, and SOCRATE works on the basic operators used as library by the code generated by the compiler. SOCRATE would be very interested in HLS tools for FPGA to apply them to radio applications, in particular for the CorteXlab platform: the platform includes FPGA, but radio applications are too complex to be written by hand in VHDL. SOCRATE could therefore be both a partner of CASH to design HLS tools, and a user of these tools.
- Hardware/software system simulation: SOCRATE works at the interface of software and hardware, and need simulation to prototype new solutions that can later be integrated in actual hardware. Matthieu Moy already co-supervises a Ph.D with SOCRATE and plans to continue the interactions with the team. SOCRATE is a user of simulation tools, while CASH will work on improving the simulation tools themselves.

In all cases, we envision mutually beneficial collaboration, but the border-line between the teams is clear, and there is very little overlap in the topic of our activities.

A.4 Synchrone (Verimag)

A.4.1 Summary of Synchrone's activities

The Synchrone team was created around the Lustre synchronous data-flow programming language. The team now works on several topics, which can be found on the following webpage:

`http://www-verimag.imag.fr/Synchrone-main?lang=fr`

The ones relevant in a comparison with CASH are:

- Languages and Tools for Critical Real-Time Systems. Synchrone works in particular on the Lustre language. The current main topics of interest are the implementation of critical applications on many-core architectures, and the analysis of worst-case execution time (WCET).
- Virtual Prototyping and Simulation. This research is done in collaboration with STMicroelectronics, and is centered on the SystemC/TLM technology for System-on-a-Chip modeling and simulation. Recent work on the topic include modeling of extra-functional properties like power-consumption and temperature and parallel execution of simulation.

A.4.2 Positioning of CASH

The Synchrone team of Verimag is currently led by Matthieu Moy. If Matthieu Moy moves to LIP next year, part of the simulation-related activities will be brought to CASH, but activities related to real-time and code generation for Lustre will not.

The common point with the Synchrone team at Verimag is the belief that the dataflow model fits particularly well the notion of task parallelism. We aim to work on the expressivity of such languages, but we desire to go beyond the synchronous paradigm. Similarly, we share the idea that working on the language or appropriate intermediate representations will facilitate further analyses and optimizations.

Different preferred applications (critical embedded programs for Synchrone, HPC for CASH) entail different constraints: contrarily to the Synchrone team, our objective is not to optimize the worst-case execution time (WCET), but the average performance. We will not compute WCETs for critical software.

The Synchrone team does not study optimizing compilers, and targets simple compiler implementation flows to keep traceability between the source code and the generated code.

A.4.3 Possible collaborations

The presence of a former member of Synchrone within CASH will obviously create a favorable context for collaboration. However, the goal of CASH is clearly not to reproduce the Synchrone team and both teams have different objectives. We may collaborate on code generation for many-core architectures and simulation.

A.5 SPADES

We share with SPADES the idea that the development as well as the efficient and safe compilation of programs is through the usage of adapted programming paradigms, such as the dataflow formalism. Their work on distributed synchronous programs is also a common point with us.

As for the Synchrone team, the application domain of SPADES is embedded systems, that have specific constraints that are different from the constraints of classic HPC. We will work neither on fault tolerance, nor on the theoretical aspect of component-based programming. SPADES does not tackle the problem of optimizing compilation for HPC applications, nor circuit synthesis, nor the design of dedicated static analyses.

A.6 PARKAS

We share with the PARKAS team the vision of a dataflow formalism for parallel programming and for the construction of compilers. Nevertheless, Marc Pouzet focuses on the resolution on front-end programming languages issues (typing, functional and higher order extensions, hybrid control and reactive systems) than on the generation of optimized code. The activities of Albert Cohen on deterministic parallel programming and the optimization of HPC kernels are complementary to ours, such as dynamic aspects in task parallelism and cross-cutting polyhedral building blocks (scheduling, code generation, embedding into a back-end compiler).

PARKAS has no activity on the design of static analyses by abstract interpretation, and does not study HLS.

A.7 PACSS (Verimag), ANTIQUE (Paris), CELTIQUE (Rennes)

A.7.1 Summary of PACSS activities

PACSS (`http://www-verimag.imag.fr/PACSS.html?lang=fr`) works on program analysis and verification[6]. It follows several directions:

- Static analysis (in particular Abstract Interpretation) to prove properties or extract invariants from programs.
- Static analysis applied to security.
- Interactive proofs (especially using the coq proof assistant).

Only the first axis is relevant in a comparison with CASH. Its main participants are David Monniaux and Nicolas Halbwachs.

[6] All the text concerning PACSS has been proof-read, discussed and validated by D. Monniaux, head of the team.

A.7.2 Summary of ANTIQUE activities

The ANTIQUE team (`https://www.di.ens.fr/AntiqueTeam.html.fr`) studies:[7]

- abstraction techniques in general (abstract interpretation of course, but also modeling techniques in biology);
- abstract domains and static analysis techniques
- applications to software verification with a wide spectrum of software kinds (synchronous, parallel, concurrent, data-structure intensive, numeric...) and properties (safety, security, liveness).
- applications to other fields, such as biology.

Only the activities concerning static analyses of general purpose programs (or safety-critical ones, with the Astree Analyzer [9]) are relevant for a comparison with CASH. The participants are Xavier Rival, Cezara Dragoi, Jerôme Féret and Vincent Danos.

A.7.3 Summary of CELTIQUE activities

CELTIQUE (`https://www.irisa.fr/celtique/`) works on programs reliability and security[8], especially in the following directions:

- Program analyses by abstract interpretation, and decision procedures.
- Formal certification of static analysis tools and compilers.
- Application to software security.

The activities on abstract interpretation and on formal semantics of intermediate representations inside compilers (SSA), are relevant for a comparison with CASH. The participants are Delphine Demange, David Pichardie, Sandrine Blazy, Fréderic Besson, and Thomas Jensen.

A.7.4 Positioning of CASH

These three teams have activities on abstract interpretation that meet ours. We share the ambition of developing expressive and usable analyzers, that scale.

Nevertheless, the work on suitable intermediate representations and the design of static analyses for optimizing compilers is only common with some researchers of the CELTIQUE team (Delphine Demange, David Cachera). The constraints of an analysis for safety or compilation are different: the former favor precision and the second has higher performance constraints. CELTIQUE has complementary activities on compilation: they share our credo that compilers should be build on solid semantic foundations, as shown by their involvement in the CompCert project, but they do not specifically develop new compiler optimizations.

Furthermore, PACSS, ANTIQUE and CELTIQUE are only focusing on software while we also target hardware verification and synthesis.

A.7.5 Possible collaborations

Both David Monniaux and Nicolas Halbwachs are former members of the Synchrone team led by Matthieu Moy up to 2017. PACSS and Synchrone still have very close interactions. David Monniaux and Matthieu Moy already co-supervised 3 internships and 1 Ph.D. David Monniaux and Laure Gonnord already co-authored three conference papers. The context is therefore very favorable to collaboration:

[7] All the text concerning ANTIQUE has been proof-read, discussed and validated by X.Rival, head of the team.

[8] All the text concerning CELTIQUE has been proof-read, discussed and validated by T. Jensen, head of the team.

- Laure Gonnord and David Monniaux have an ongoing work on designing array abstractions, and a first paper in SAS ([16]) which is strongly related on the fundamental research on computation dependencies of the CASH project.
- Since January 2017, we organize a common weekly "Reading Group", via teleconference, on the static analysis topic[9].

We do not currently have any projects with ANTIQUE, but we could for example on topics related to the Anastasec ANR project[10], whose goal is the automatic proof of security properties on low level codes, which could be instantiated in our particular context of HLS, and for which a deep work on scalability is imperative.

We do not not share any projects with CELTIQUE. With Delphine Demange and David Pichardie we could collaborate on developing certified compiler optimizations, as a medium-term project.

A.8 Inria teams of the theme "Architecture, languages, compilation"

A.8.1 CAMUS

CAMUS targets the conception of a production chain for efficient execution of an application. CAMUS has an activity on static parallelization and optimization, which is complementary with dynamic parallelization via profiling and a work on speculative models. We will not study dynamic compilation and speculative models, nor runtime systems. Rather, we will try to focus on the static part of compilation and synthesis.

CAMUS has also an activity on the formal proof of formal transformations, that we will not work on. CAMUS focus on multicore architectures, and do not target HLS.

A.8.2 PACAP

The PACAP team works on two topics related to ours: timing analysis for embedded systems (Worst-case Execution Time, WCET, analysis and scheduling) and dynamic compilation. We plan to use WCET-related techniques in our analysis (for load balancing for example), but unlike PACAP we do not target embedded or critical systems as a target. Regarding the compilation activities, the fundamental distinction is the focus on static techniques of CASH.

A.8.3 CAIRN

CAIRN's activities on hardware synthesis and compilation is close to ours. We will share a common interest for polyhedral-based optimization, but unlike CAIRN we will not use the reconfigurability power of FPGAs and we plan to address irregular applications. We will also target HPC applications as opposed to embedded systems.

Summary of CAIRN's activities CAIRN's activity report is available here:

`http://raweb.inria.fr/rapportsactivite/RA2016/cairn/cairn.pdf`

The overall objective of CAIRN is to study reconfigurable architectures as an energy efficient computing paradigm. Its main activities are:

- The design of new reconfigurable architectures with an emphasis on flexible arithmetic, dynamic reconfiguration management and low power consumption.

[9] `http://stator.imag.fr/w/index.php/VerifGroup`
[10] `https://www.di.ens.fr/~feret/anastasec/`

- The design of dedicated synthesis algorithms and compiler optimizations.
- Applications to wireless networks and cryptography.

Only the second direction is relevant for CASH, we do not plan to study dynamic reconfiguration management. The activities of CAIRN in this direction are:

- Polyhedral Loop Transformations for High-Level Synthesis. The activity focuses on the design of source-level transformations for HLS based on the polyhedral model. A particular aspect of this activity is to improve the efficiency and applicability of nested loop pipelining. CAIRN develops a compiler toolbox called Gecos which capitalizes all its developments. Members: Steven Derrien, Patrice Quinton, Tomofumi Yuki.
- Regular Processor Arrays. The activity focuses on the mapping of regular kernels with uniform dependencies (represented as systems of recurrence equations) to parallel processor arrays architectures. This research aims at developing methods and tools to automate the synthesis of such architectures for data-intensive applications. Systolic array synthesis is one of the fundamental seminal work of the polyhedral model, for which CAIRN members have a strong competence. Members: Steven Derrien, Patrice Quinton, Tomofumi Yuki.

Positioning of CASH CAIRN's project intersects ours mostly with Section 1.3 for which future collaborations are possible. Though static analysis and HPC compilation techniques are progressively disseminating in the HLS community, many locks have to be overcome, for which a single team is clearly not sufficient. CAIRN's research on HLS leverages polyhedral techniques to design source-level optimization for HLS. CASH will rely on direct hardware implementation to validate specific hardware mechanisms (e.g. synchronizations). But for system-level design, CASH will produce source-to-source program transformations in front of existing HLS tools. For regular applications, CASH will rely on polyhedral techniques. We believe that a collaboration is possible on this point. Unlike CAIRN, CASH will study how irregular applications can be mapped to reconfigurable architectures. Also, CASH does not plan to leverage the dynamic reconfigurability of FPGA.

A.9 System-Level Synthesis (TIMA)

The SLS team of TIMA is know for its expertise in simulation, in particular using SystemC/TLM. The SLS team focuses on techniques to integrate embedded software in the platform: dynamic binary translation and native simulation. CASH does not plan to work on these subjects. Also, SLS targets simulations with precise timing, while CASH plans to work on loosely timed simulations, which raise completely different issues especially when it comes to parallelization [7].

SLS also focuses on high-level synthesis for HPC and embedded systems targeting FPGA. Their tool, AUGH, relies on design-space exploration associated with fast hardware synthesis [22]. CASH targets program optimizations based on the polyhedral model, which are more accurate, but more expensive; hence less adapted to design space exploration. Thus our approaches are complementary. Also, SLS focuses on HLS-level system mechanisms for FPGAs. Recently, they developed a context switch on hardware tasks executed on reconfigurable systems [10]. CASH does not plan to work on these subjects. Our purpose is to focus on resource efficient synthesis of compute-intensive kernels.

A.10 Département Architectures Conception et Logiciels Embarqués (CEA LIST-LETI)

CEA-LIST works on simulation with SystemC/TLM, with an emphasis on many-core processor, and parallel/heterogeneous simulation. We already have a funding for a joint Ph.D and submitted a European project together, and plan to continue collaboration.

A.11 International projects and teams

A.11.1 High-Level Synthesis (HLS)

- The **VAST** laboratory (VLSI Architecture, Synthesis, and Technology) led by Prof. Jason Cong from University of California Los Angeles targets customized computing for big data applications, energy-efficient computing and electronic design automation. VAST has many achievement, including source-level optimimations for FPGA: data transfert optimization, parallelization or multibank memory allocation to quote a few. CASH shares the same goals than VAST, but with a slightly different approach. Our goal is rather to design dataflow intermediate representation tailored to the requirements of reconfigurable computing with the corresponding hardware generation algorithms. We believe that a collaboration is possible either on source-level polyhedral analysis or on low-level hardware generation.
- **ROCCC** (Riverside optimizing compiler for configurable computing) is an HLS project led by Prof. Walid Najjar from University of California Riverside, USA. It features compiler optimizations as parallelism extraction, data reuse or design space exploration. Though the project is more oriented to embedded systems, many ideas could profit to HPC, as their hardware mechanism to fetch data for sliding window loop nest (smart buffer) or to optimize irregular data accesses. We believe that a collaboration is possible on the hardware generation part.
- The **Compaan** project was initially led by Prof. Ed Depreterre, Bart Kienhuis, and Todor Stefanov from University of Leiden, Netherlands. It aims at providing an HLS tool for accelerating compute-intensive embedded applications. The compaan project came up with the interesting concept of Polyhedral process network (PPN), a dataflow representation tailored to regular kernels. CASH shares the idea that dataflow representations are fundamental to design reusable HLS analysis and optimizations. However, the PPN model is too restricted for our purpose. We believe that a fructfull collaboration is possible on the dataflow model part.
- The **Circuits and Systems group** led by George A Constantinides from Imperial College, London, aims at studying architectures, synthesis tools, and applications of customised hardware. There achievements range from hardware mechanisms to HLS optimizations. For example, they proposed a source-level data transfert optimization. They also investiguate HLS for irregular applications. We think a collaboration is possible for most of the issues we plan to investiguate.
- The **Panda** project led by Fabrizio Ferrandi from Politecnico di Milano (Italy) investiguates HLS specific issues concerned with parallelism extraction, hardware/software partitioning and mapping, metrics for performance estimation of embedded software applications and dynamic reconfigurable devices. CASH does not target embedded systems. However, we believe that a collaboration is possible on source-level analysis and optimizations.

A.11.2 Simulation of Systems-on-a-Chip

- The **group of computer architecture at university of Bremen** works among other topics on simulation, in particular SystemC. Initially, the team worked essentially on low levels of abstraction like gate-level or Register Transfer Level (RTL). Activities with high-level models like TLM (which is the scope of CASH) target tools to help the programmer understand a program (fault localization, visualization) and program verification. The first topic is not in the scope of CASH, but we may collaborate on the second.
- The **Institute for Communication Technologies and Embedded Systems (ICE), Aachen University** has activities very close to the simulation axis of CASH: they also work on parallel execution

of SystemC/TLM programs. However, their work target models with fine-grained timing, and we showed that this category of approach is not applicable to the models we target in CASH [7].

A.11.3 Static analyses groups in Europe/US

(Note: this section is work in progress and will be detailed in the final version)

The following teams work among other topics on abstract interpretation:

- UK: Oxford (Kroening), Kent (King)
- Germany: Freiburg (Podelski)
- Austria: Wien (Kovacz, Sinn)
- Spain: IMDEA Madrid (Hermenegildo, Gallagher)
- Italy: Verona (Giacobazzi), Pescara (Scozzari), Parma (Zaffanella)
- Danemark (Nielsen)
- Switzerland: ETHZ (Vechev), EPFL (Kuncak)
- US: Facebook (Logozzo), NYU (Cousot), Wisconstin (Reps)

B Details on Research Group

We propose Matthieu Moy as team leader. Matthieu was "maître de conférences" at Verimag, Grenoble, and was recruited with the same status at Université Claude Bernard, Lyon 1 (UCBL) since September 2017.

- **Matthieu Moy**. `https://matthieu-moy.fr`. Matthieu received his Ph.D degree in Computer Science from Grenoble INP in 2005. During his Ph.D, he applied formal verification on models of Systems-on-a-Chip written in SystemC, as part of a collaboration with STMicroelectronics. After he joined Verimag/Ensimag as an assistant professor (his current position), he continued working with STMicroelectronics on various topics including formal verification, compilation techniques for SystemC, modeling of energy consumption and parallel execution of simulations. He also worked on abstract interpretation and real-time calculus. Recently, his main research interest moved to critical systems on many-core architecture, including code generation from synchronous languages and timing analysis. He obtained his Habilitation (HDR) in 2014 from Grenoble INP, and became the leader of the "Synchrone" team in Verimag in 2015. He co-supervised 2 full Ph.Ds, and is now supervising 4 Ph.Ds (including 3 as main supervisor).
- **Christophe Alias**. `http://perso.ens-lyon.fr/christophe.alias`. Christophe received his PhD degree in Computer Science from University of Versailles in 2005. He is currently a research associate (CR1) at INRIA. His research interests focus on compilers for high-performance computing, with a strong emphasis on high-level-synthesis of circuits with polyhedral techniques. He worked on many subjects around HPC compilers, including array SSA, vectorization, I/O optimization, buffer allocation, or dataflow models for circuit synthesis. He co-supervised 2 Ph.D students. He co-founded the XtremLogic startup, which exploits the parallelizing compiler he wrote to synthesize circuits on FPGA.
- **Laure Gonnord**. `http://laure.gonnord.org/pro/`. Laure received her PhD degree in computer science from the University Joseph Fourier (Grenoble), in 2007. She has been an assistant professor at the University of Lille, and currently holds an assistant professor position at UCBL. Her main research interests lie in the design of static analyses, with emphasis on the automatic synthesis of numerical invariants and application in compilation (scheduling) and termination proofs. She has experience in the development of static analyses for synchronous programs. She has already

published in major conferences on static analysis or compilation.